

AD-A158 367

BULK CMOS VLSI TECHNOLOGY STUDIES PART 1 SCALABLE CMOS  
DESIGN RULES PART 2. (U) MISSISSIPPI STATE UNIV  
MISSISSIPPI STATE DEPT OF ELECTRICAL E.

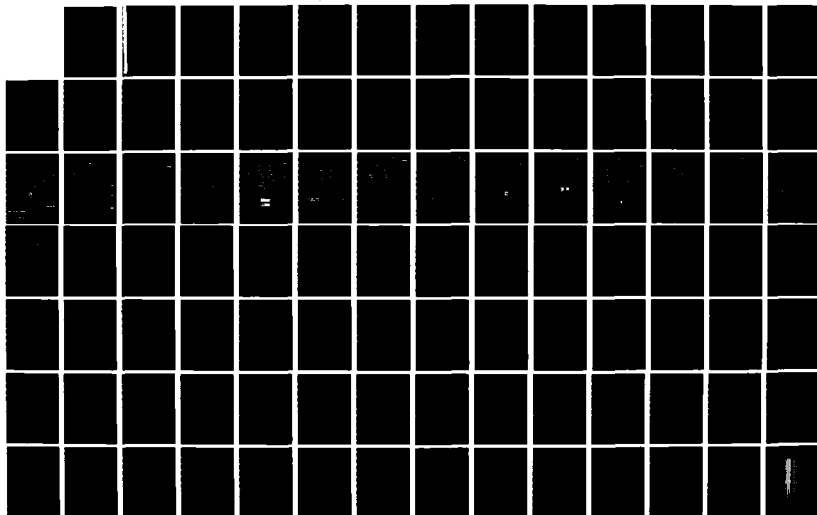
1/3

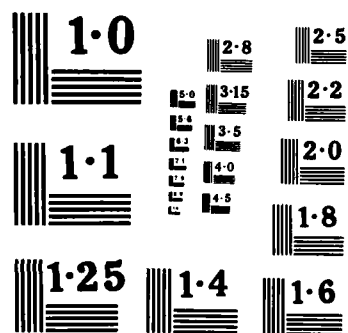
UNCLASSIFIED

J D TROTTER ET AL. 17 JUN 85

F/G 9/5

NL





NATIONAL BUREAU OF STANDARDS  
MICROCOPY RESOLUTION TEST CHART

①

AD-A158 367

FINAL REPORT

CONTRACT DAAG29-82-K-0167

BULK CMOS VLSI TECHNOLOGY STUDIES

PART 1: SCALABLE CMOS DESIGN RULES

PART 2: CMOS APPROACHES TO PLA DESIGN

Principal Investigator

J. Donald Trotter

Associate Investigator

Ajay Kumar Reddy Naini

Mississippi State University  
Department of Electrical Engineering  
Mississippi State, Mississippi 39762

for  
Defense Advance Research Projects Agency  
1400 Wilson Ave.  
Arlington, VA 22209

for  
U. S. Army Research Office  
P. O. Box 11211  
Research Triangle Park, NC 27709

June 17, 1985

DTIC FILE COPY

DTIC  
SELECTED

AD-A158 367

1

This document has been approved  
for public release and sale; its  
distribution is unlimited.

85 8 28 038

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

ADA 158 367

# REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT <div>This document has been approved for public release and sale; its distribution is unlimited.</div>	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		7a. NAME OF MONITORING ORGANIZATION Georgia Institute of Technology	
6a. NAME OF PERFORMING ORGANIZATION Miss. State University	6b. OFFICE SYMBOL (If applicable) Electrical	7b. ADDRESS (City, State and ZIP Code) 206 O'Keefe Building Atlanta, GA 30332	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION U.S. Army Research Office		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER DAAG29-82-K-0167	
8b. ADDRESS (City, State and ZIP Code) P.O. Box 12211 Research Triangle Park, NC 27709-2211		10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. PROJECT NO. TASK NO. WORK UNIT NO.	
11. TITLE (Include Security Classification) Bulk CMOS VLSI Technology Studies		12. PERSONAL AUTHOR(S) J. Donald Trotter, Ajay Kumar Reddy Naini	
13a. TYPE OF REPORT Final	13b. TIME COVERED FROM 8-82 TO 2-85	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
16. SUPPLEMENTARY NOTATION			
17. CCSATI CODES FIELD GROUP SUB. GR.		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Part 1: Scalable CMOS Design Rules Part 2: CMOS Approaches to PLA Designs	
19. ABSTRACT (Continue on reverse if necessary and identify by block number) Part 1: Scalable CMOS design rules are developed for the MOSIS community to facilitate fabrication from a single design at 3 microns and 1.2 microns VHSIC dimensions. Part 2: Various Programmable Logic Array (PLA) implementations with clocked CMOS technology are explored in this project. Three different CMOS PLA circuit styles are described: the "large" PLA uses a gated OR plane and is useful for a system with large number of inputs; the "moderate" PLA and the "small" PLA are ripple varieties with the former having the capability of handling a larger number of inputs than the latter. Path Programmable Logic (PPL), which is a folded form of a PLA, is also studied. A symbolic form of representation is developed and future PPL development activities are discussed. The PPL approach has a size and flexibility advantage over the other PLA approaches - except in applications requiring large PLA's.			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS <input type="checkbox"/>		21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL		22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL

DD FORM 1473, 83 APR

EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

FINAL REPORT

CONTRACT DAAG29-82-K-0167

BULK CMOS VLSI TECHNOLOGY STUDIES

PART 1: SCALABLE CMOS DESIGN RULES

PART 2: CMOS APPROACHES TO PLA DESIGN

Principal Investigator

J. Donald Trotter

Associate Investigator

Ajay Kumar Reddy Naini

Mississippi State University  
Department of Electrical Engineering  
Mississippi State, Mississippi 39762

for  
Defense Advance Research Projects Agency  
1400 Wilson Ave.  
Arlington, VA 22209

for  
U. S. Army Research Office  
P. O. Box 11211  
Research Triangle Park, NC 27709

June 17, 1985

This document has been approved  
for public release and sale; its  
distribution is unlimited.

✓  
*per Para 17 on file.*

A-1

TABLE OF CONTENTS  
FINAL REPORT  
CONTRACT DAAG29-82-K-0167  
BULK CMOS VLSI TECHNOLOGY STUDIES

<u>Volume I</u>	Page
Part 1: Scalable CMOS Design Rules	
Chapter 1: Introduction to Final Report . . . . .	1
Chapter 2: Scalable CMOS Design Rules . . . . .	4
Part 2: CMOS Approaches the PLA Design	
Chapter 1: Introduction . . . . .	1
Chapter 2: Basic CMOS Circuit Concepts . . . . .	4
Chapter 3: Large PLA . . . . .	12
Chapter 4: Ripple PLA's . . . . .	51
Chapter 5: Path Programmable Logic . . . . .	73
Chapter 6: Conclusions . . . . .	99
Appendix A: Pascal Program . . . . .	105
Appendix B: Scalable Design Rules . . . . .	115
Appendix C: PLA's Cell Library . . . . .	131
Appendix D: PPL Cell Set . . . . .	159
<u>Volume II</u>	
Part 3: A 1.2 Micron CMOS Data Path Chip	
Chapter 1: Introduction . . . . .	1
Chapter 2: Data Path . . . . .	13
Chapter 3: Microword Format and Control . . . . .	28
Chapter 4: The Frame . . . . .	51
Chapter 5: Functional Testing . . . . .	61
Chapter 6: System Example . . . . .	73
Chapter 7: Second Generation Enhancements . . . . .	84

Chapter 8: Conclusion . . . . .	95
Appendix . . . . .	100

### Volume III

#### Part 4: Design of a CMOS Microsequencer

Chapter 1: Microsequencer Overview . . . . .	1
Chapter 2: Functional Description of Microsequencer Macros . . . . .	15
Chapter 3: Microsequencer Circuit Styles and Layouts . . . . .	36
Chapter 4: Functional Testing of the Microsequencer . . . . .	64
Chapter 5: A Next Generation Microcontroller . .	69
Appendix A: Functional Description of the Microsequencer . . . . .	74
Appendix B: Logic Descriptions and Circuit Schematics of Macros . . . . .	88
Appendix C: Cell Library . . . . .	122

### Volume IV

#### Part 5: The Design and Implementation of a High-Speed Integrated Circuit Functional Tester

Chapter 1: Introduction . . . . .	1
Chapter 2: Hardware Design . . . . .	8
Chapter 3: Software Design . . . . .	40
Chapter 4: Operation . . . . .	51
Chapter 5: Evaluation and Discussion . . . . .	61
Appendix A: Test Station . . . . .	67
Appendix B: Functional Tester Schematics . . . .	69
Appendix C: Assembly Language Programs . . . . .	101
Appendix D: Pascal Program . . . . .	115
Appendix E: Test Results . . . . .	168

PART I  
SCALABLE CMOS DESIGN RULES  
TABLE OF CONTENTS

	Page
Chapter 1. INTRODUCTION TO FINAL REPORT . . . . .	1
Chapter 2. SCALABLE CMOS DESIGN RULES . . . . .	4
2.1 Objective of Scalable Rules . . . . .	4
2.2 The Approach with MOSIS . . . . .	5
2.3 The Challenges and Issues . . . . .	5
2.4 CMOS Design Rules for Lambda-Based Designs . . . . .	10
2.5 Design Rule Calculations for CMOS Design Rules at LAMBDA = .8 and 1.5 Microns . . . . .	13
2.6 Design Rule Calculations For CMOS Design Rules at LAMBDA = .7 and 1.2 Microns . . . . .	16
2.7 The MOSIS Scalable Design Rules for Students . . . . .	19



## Chapter 1

### INTRODUCTION TO FINAL REPORT

This research focuses on the issues relevant to designing practical high performance CMOS VLSI circuits utilizing the 1.2 micron CMOS technology obtained from MOSIS. As such the research has supported the following activities:

- 1) the vendor interface for MOSIS,
- 2) the development of design rules,
- 3) the characterization of the vendor processing,
- 4) the development of the CAD tools for the designers,
- 5) the development of circuit methodologies appropriate for CMOS microcomputers, and
- 6) the development of canaries, experimental circuits which demonstrate the circuit and design methodologies.

Much of the information relative to the vendor interface and the individual vendor process characteristics is company proprietary. After the envelope of process characteristics from the various vendors is developed by MOSIS, this appropriate information is provided to the community directly by MOSIS. Consequently, this report will address the research relative to CMOS circuit development.

The concept of scalable CMOS design rules to facilitate fabrication of the design at current production technologies down to advanced VHSIC geometries is developed in the next chapter. These design rules are applicable for service through the MOSIS system with MOSIS providing the processing of the design file to translate to the actual mask dimensions used in the pattern generator. The intent is for MOSIS to provide, in the future characterization data, for the different technologies supported. This characterization is to be developed under other funding and implemented in both the absolute and lambda-based terms.

The design rules are presented in the "user-friendly" version distributed by MOSIS and in the compact style with the extensive calculations as developed at MSU under this contract.

Previous research at MSU has dealt with clocked CMOS circuit styles with some application to gate array and microprocessor applications. Work under this contract has allowed these concepts to be developed in the full custom environment, but focuses on the requirements in implementing custom microprocessors.

The first application area presented in this report deals with structured logic schemes based on Programmable Logic Arrays (PLAs). Three different PLA design methods are reported with a comparison of the differences. In addition, a structured, but flexible, version of the PLA, in the form of the PPL (Programmable Path Logic) is developed. Appreciation is expressed to Kent Smith, et al, of the University of Utah for their cooperation in the PPL research. The MSU version utilizes clocked CMOS circuitry in its implementation, although a ripple signal is propagated through the logic between the state latches. This clocked version is approximately four times smaller than that of the static CMOS version first obtained at Utah. The structured logic circuit development is aimed at an efficient method of generating the "glue" logic around the other major functional blocks used in the design of microcomputers.

The issues relative to the design of a CMOS data path are dealt with in the next part of the report. Clocked CMOS circuitry is developed with material presented in a tutorial form for students. The concept of a N-mostly design is developed in comparison to a more conventional CMOS circuit style. The N-mostly circuit approach utilizes PMOS for latches, buffers, and precharge devices with transfer gates and most logic circuitry implemented in NMOS. The N-mostly approach saves considerable area in the circuitry associated with transferring data over the bus, through the shifter, and into the ALU input latches. The full data path canary, developed under this research effort, is implemented using single layer metal and conventional CMOS transfer gates. It is designed with the original 1.2 micron design rules which are similar but not compatible with the scalable set. Another experimental circuit has been designed to test the N-mostly approach to the bus transfers.

The data path report also includes the development of a pad frame with I/O circuitry to be used by other designers with the 1.2 micron technology. Although this work preceded the released MOSIS version of the scalable design rules and is not "scalable", present work with MOSIS is leading to a redesigned padframe which is compatible with the scalable concept. The circuit style in this report is carried over to the scalable version to be released in the future by MOSIS.

A microsequencer designed for CMOS is developed in the next part of the report. This microsequencer is an enhancement of the Mead and Conway version and is similar to the AMD 2900 series. An experimental canary based on this research is being reworked to incorporate the new scalable design rules, again under separate funding. This microsequencer is expected to become the heart of a control

unit to be implemented in the scalable rules. Our plans at MSU are to incorporate in the future this basic microsequencer with two small ALUs for control of external memory and internal data memory to form a fairly powerful control unit, except for the programmable control store.

The final part of the report deals with the development of a tester, suitable for university researchers to evaluate their experimental canaries. The tester allows the downloading of test vectors from a HP9820 through a 16 bit port into dedicated memory for testing. This memory, which is modular in construction with 8 channels per module, facilitates testing at speeds up to a 10 MHz clock rate. The device under test (DUT) outputs are loaded back into the tester memory in real time and evaluated off-line after being uploaded into the HP9820. Fairly simple control of the logic levels at the DUT interface are provided. This arrangement minimizes the complexity of the circuitry involved but is suitable for university research.

## Chapter 2

### SCALABLE CMOS DESIGN RULES

#### 2.1 Objective of Scalable Rules

The objective of developing scalable CMOS design rules is two fold: 1) to facilitate translating a design data base from one CMOS technology to another technology transparently, and 2) to fabricate more economically student projects or other experimental designs, first, at the 3 micron technology and, later, at the 1.2 micron technology.

The MOSIS system of supplying fabrication service primarily achieves its cost effectiveness by spreading the fabrication cost over many different designs. The fabrication masks contain, maybe, 50 to 200 different designs for each wafer. The tooling (masks) and wafer fabrication cost of approximately \$25,000 is spread over the 50 or more designs to achieve less than \$500 fab cost for each individual design. Utilizing full wafer exposure allows the entire wafer area to be unique. In contrast, standard production procedures commonly utilize several hundred patterns which are repeats of a single design.

At the 1.2 micron technology, full wafer exposure is prohibitively expensive with direct e-beam or X-ray lithography being the candidates. For production requirement reasons, the silicon foundries emphasize wafer steppers which provide only approximately one square centimeter of unique pattern area, which is repeated across the wafer. This limited pattern area limits the number of designs which can absorb the rather high fabrication cost. The results is that prototyping at the 1.2 micron level will result in fabrication cost at least 10x over the 3 micron fab cost.

At least for the learning exercises and initial debug phases of design, it is desirable for a designer to be able to experiment at the more cost effective CMOS technology and proceed to the high performance technology as justified. Scalable design rules permit the regeneration of the required artwork without extensive designer rework.

## 2.2 The Approach with MOSIS

The concept of the scalable rules evolved from our work at Mississippi State University to facilitate both the teaching of CMOS circuit design and the research effort relevant to the 1.2 micron program. In order to generate experimental structures with various design rules, we developed our own software GRAIL with a technology table of feature sizes which are pointed to from the design data base. By changing the scale factor of the grid system and the technology table, a design can be run at different technologies. The feature sizes are only loosely tied to the scaling factor, thereby, permitting non-linear scaling and compensations as needed for the particular technology and particular vendor. GRAIL provides the flexibility of a symbolic design system with the readability of conventional mask drawings. This system is unique to MSU, however, and is not easily transported.

MOSIS requested a scheme (system) with the following properties:

- 1) It should not require any additional software by the designer -- thus ruling out symbolic methods.
- 2) It should be as simple as possible for the convenience of the design community, but compatible with all of the existing design methods used by the MOSIS community.

MOSIS would accept a rather coarse grid design which more-or-less represented the design (symbolic?); MOSIS would then "operate" on the design file with MOSIS software which would provide the translation to high resolution masks for a particular technology. Simple!

## 2.3 The Challenges and Issues

Reviewing the different design methods used in the community and certain constraints from the MOSIS environment revealed very interesting trade-offs, too many to elaborate on. However, the general problem is worthy of discussion.

First, in spite of what is drawn (or entered in the CIF design file which is submitted to MOSIS) with a supposed resolution of .01 microns, the masks are generated with feature edges falling on increments (drawing units) of a much lower resolution. This drawing unit corresponds to the "spot" size of the electron-beam used in the pattern generation process. The first trade-off involves the size of the beam (spot size). The smaller the spot, the finer the grid which defines the edges of all of the graphic features and the higher the mask generation costs. Reducing the spot by a factor of two results in four times the masks.

cost. The MOSIS constraint: use as large a spot size as possible.

The largest spot size available is slightly over one micron -- on the mask. At the 3 micron technology, it is common to use an  $1/2$  micron beam size -- thus providing a drawing unit of  $1/2$  micron since the mask pattern is commonly  $1\times$  compared with the wafer pattern. At the 1.2 micron technology, wafer steppers are commonly used with two different lens:  $5\times$  and  $10\times$  reduction. With the  $5\times$  lens and  $10\times$  lens, the maximum beam size of 1 micron corresponds to drawing units on the wafer of 0.2 micron and 0.1 micron, respectively. Considering the wider angle of view available at  $5\times$  and the lower cost of the mask per design, the  $5\times$  is preferred if the image resolution on the wafer is satisfactory.

A careful researcher observes that the desire to provide flexibility to accommodate scaling leads to small drawing units which is in conflict with the desire to have a coarse grid for designer convenience. The solution appears to be the creation of two grids, a coarse grid for locating features and a finer grid for defining the edges of the features. This is the essential ingredient of the GRAIL system at MSU. The coarse grid provides simplicity for feature location; the fine grid provides the feature sizing required for adjusting the scaling for the particular technology or vendor. A necessary condition is that the fine grid (drawing units) must fall on the coarse grid ( $\lambda$ ).

The NMOS  $\lambda$  rules, illustrated by Mead and Conway, underscore the simplicity of designing with the coarse grid. Applying this concept to the common design rules for the 3 micron technology leads to a  $\lambda$  of 1.5 microns. However, one observes that process engineers have generally required greater feature linewidths and spaces for the second metal interconnect and vias than for the underlying structures. As a result, one is forced either to represent the upper interconnect structure with more grid units than for the substructure or to represent the interconnect structures similarly with a compromised grid size and provide suitable adjustment in feature sizes. There is little question in the author's mind that the latter approach is far easier on the designer for comparable layout densities. For that reason the author is convinced that the  $\lambda$  grid size should be based on the interconnection representation concerns and not on simply twice the minimum gate length as utilized by Mead and Conway. The gate length is 1.2 microns in the 1.2 micron technology, and this argument suggest that the  $\lambda$  grid should be something greater than the 0.6 microns. Considering the drawing unit to be 0.2 microns for this 1.2 micron technology, the next increment for the  $\lambda$  grid becomes 0.8 microns.

**CMOS  
SCALABLE  
DESIGN RULES**

**MAY 1985**

## 2.7 The MOSIS Scalable Design Rules for Students

The following pages are non-colored copies of the MOSIS-provided version of the scalable design rules. One should contact MOSIS directly for a copy of the official colored version.



## RULES UNDER CONSIDERATION BUT NOT RELEASED

XSA	4	1.225	1.925	2.75	2	3.2	2.67
GSDs	3	1.225	1.4	2	2	2.4	2.00
DSXb	3	-.175	.2625	.375	-.4	.4	.33
PSG & POG	2	1.4875	1.575	2.25	2.6	2.8	2.33
FW	2	2.1	1.575	2.25	3.6	2.7	2.25
AOV	1	.9625	.4375	.625	1.8	.9	.75
VSC	1	.7875	.525	.75	1.4	.9	.75
SSe	3	1.4	2.1	3	2.4	3.6	3.00
DOCm*	-.5	.7	.525	.75	1.4	1	.83
FOCm*	.5	.7875	.4375	.625	1.4	.75	.62
FSe	2.5	1.05	1.575	2.25	1.8	2.7	2.25
VSA	1.5	.7875	.9625	1.375	1.2	1.5	1.25
VSG	1.5	1.1375	1.05	1.5	2	1.9	1.58
GOV	1.5	.9625	.7	1	1.6	1.1	.92
VOC2	.5	.4375	.525	.75	.8	.9	.75
SW	2	2.1	1.4	2	3.6	2.4	2.00

- NOTE:
- >The "e" suffix indicates the case where two interconnect lines approach each other at their "ends", the ends being no greater than 4 lambda across.
  - >The "s" suffix indicates a short 4 lambda feature which has this minimum space to another feature.
  - >The "m\*" suffix applies in a direction not in the direction of the metal interconnect line.
  - >The VOC2 applies in the case of a via stacked over a contact with the requirement that at least two edges of the via overlap the contact by VOC2.
  - >The "b" suffix indicates the special case for a modified butting contact structure between dogbone poly and active with two contact cuts and jumpered with first metal.

P	BLOAT		0	0 align >A	0	0 align >A		
PSG & POG		3	2.1875	2.275	3.25	3.8	4	3.33
POA		2	1.1375	1.4875	2.125	1.8	2.4	2.00
POX		3	.7875	1.1375	1.625	1.4	2	1.67
PSA		2	1.1375	1.4875	2.125	1.8	2.4	2.00
PSX		3	.7875	1.1375	1.625	1.4	2	1.67
POCw&PSCw		1	.7875	.7	1	1.4	1.2	1.00
C	BLOAT		-.175	.175 align >A	-.4	.4 align >G		
CW		2	1.225	1.4	2	2	2.4	2.00
CS		2	1.575	1.4	2	2.8	2.4	2.00
CSG		2	1.575	1.575	2.25	2.8	2.8	2.33
DOC		0	1.05	.875	1.25	2	1.6	1.33
XOC, XOCw		0	1.4	.9625	1.375	2.4	1.6	1.33
F	BLOAT		.7	-.525 align >C	1.2	-.9 align >C		
FW		3	2.8	2.275	3.25	4.8	3.9	3.25
FS		3	1.4	1.925	2.75	2.4	3.3	2.75
Fp		6	4.2	4.2	6	7.2	7.2	6.00
FOC		1	1.1375	.7875	1.125	2	1.35	1.13
V	BLOAT		0	.35 align >F	0	.6 align >F		
VW		2	1.4	1.75	2.5	2.4	3	2.50
VS		2	1.4	1.05	1.5	2.4	1.8	1.50
FOV		1	1.05	.6125	.875	1.8	1.05	.87
VSC		2	1.4875	1.225	1.75	2.6	2.1	1.75
GOV		2	1.3125	1.05	1.5	2.2	1.7	1.42
VSG		2	1.4875	1.4	2	2.6	2.5	2.08
DOV		1	1.6625	1.4	2	3	2.5	2.08
VSD		3	1.1375	1.05	1.5	1.8	1.7	1.42
VSA		2	1.1375	1.3125	1.875	1.8	2.1	1.75
AOV		2	1.6625	1.1375	1.625	3	2.1	1.75
VSX		3	.7875	.9625	1.375	1.4	1.7	1.42
XOV		0	1.3125	.7875	1.125	2.2	1.3	1.08
S	BLOAT		.7	-.7 align >V	1.2	-1.2 align >V		
SW		3	2.8	2.1	3	4.8	3.6	3.00
SS		4	2.1	2.8	4	3.6	4.8	4.00
Sp		7	4.9	4.9	7	8.4	8.4	7.00
SOV		1	1.05	.525	.75	1.8	.9	.75

2.6

DESIGN RULE CALCULATIONS  
FOR CMOS DESIGN RULES  
AT LAMBDA = .7 and 1.2 MICRONS

revision 1.2.5

VARIABLE LABEL	LAMBDA RULES	"1.2 micron" TECH			"2.4 micron" TECH		
		ILAMBDA= as drawn	.7 microns mask dim microns	1.2 microns fin.dim microns	ILAMBDA= 4 spots lambda	1.2 microns mask dim microns	3 spots fin.dim microns
W	BLOAT		0	0		-4.8	4.8
WW		6	4.2	4.2	6	2.4	6.00
WS		6	4.2	4.2	6	12	6.00
A	BLOAT		.525	-.7 align >W		1.2	-1.2 align >W
AW		2	1.925	1.225	1.75	3.6	2.4
AS		3	1.575	2.275	3.25	2.4	3.6
Ap		5	3.5	3.5	5	6	6
ApIWn		6	3.9375	4.2875	6.125	9	7.2
AnIWp		6	3.9375	4.2875	6.125	4.2	7.2
ApOWp		4	2.5375	2.8875	4.125	1.8	4.8
AnOWn		4	2.5375	2.8875	4.125	6.6	4.8
X	BLOAT		2.625	-.7 align >W		4.4	-1.2 align >W
XS		6	1.575	2.275	3.25	2.8	4
XpIWn		7	3.5875	3.9375	5.625	8.6	6.8
XnIWp		7	3.5875	3.9375	5.625	3.8	6.8
XpOWp		3	3.4125	3.0625	4.375	3.4	5.2
XnOWn		3	3.4125	3.0625	4.375	8.2	5.2
XSA		5	1.925	2.625	3.75	3.2	4.4
G	BLOAT		-.175	-.175 align >A		-.4	-.4 align >A
GW		2	1.225	1.05	1.5	2	1.6
GS		2	1.575	1.75	2.5	2.8	3.2
Gp		4	2.8	2.8	4	4.8	4.8
GSD		4	1.925	2.1	3	3.2	3.6
GOA		2	1.05	1.3125	1.875	1.6	2
AOG		2	1.75	1.4875	2.125	3.2	2.8
GSA		1	.525	.9625	1.375	.8	1.6
GSX		3	.875	1.3125	1.875	1.6	2.4
GSXs		2	.175	.6125	.875	.4	1.2
D	BLOAT		1.925	-.175 align >A		3.6	-.4 align >A
DS		5	1.575	1.75	2.5	2.4	2.8
DSA		3	.875	1.3125	1.875	1.2	2
DSAs		2	.175	.6125	.875	.0000000	.8
DSX		4	.525	.9625	1.375	.8	1.6

RULES UNDER CONSIDERATION				BUT NOT RELEASED			
XSA	4	1.4	2.2	2.75	2.5	4	2.67
GSDs	3	1.4	1.6	2	2.5	3	2.00
DSXb	3	-.2	.3	.375	-.25	.75	.50
PSG & POG	2	1.7	1.8	2.25	3	3.25	2.17
FW	2	2.4	1.8	2.25	4.5	3.375	2.25
AOV	1	1.1	.5	.625	2.25	1.125	.75
VSC	1	.9	.6	.75	1.5	.875	.58
SSe	3	1.6	2.4	3	3	4.5	3.00
DOCm*	-.5	.8	.6	.75	1.25	.75	.50
FOCm*	.5	.9	.5	.625	1.5	.6875	.46
FSe	2.5	1.2	1.8	2.25	2.25	3.375	2.25
VSA	1.5	.9	1.1	1.375	1.5	1.875	1.25
VSG	1.5	1.3	1.2	1.5	2.25	2.125	1.42
GOV	1.5	1.1	.8	1	2.25	1.625	1.08
VOC2	.5	.5	.6	.75	.75	.875	.58
SW	2	2.4	1.6	2	4.5	3	2.00

NOTE: >The "e" suffix indicates the case where two interconnect lines approach each other at their "ends", the ends being no greater than 4 lambda across.

>The "s" suffix indicates a short 4 lambda feature which has this minimum space to another feature.

>The "m\*" suffix applies in a direction not in the direction of the metal interconnect line.

>The VOC2 applies in the case of a via stacked over a contact with the requirement that at least two edges of the via overlap the contact by VOC2.

>The "b" suffix indicates the special case for a modified butting contact structure between dogbone poly and active with two contact cuts and jumpered with first metal.

P	BLOAT		0	0 align >A	0	0 align >A	
PSG & POG	3	2.5	2.6	3.25	4.5	4.75	3.17
POA	2	1.3	1.7	2.125	2.25	3	2.00
POX	3	.9	1.3	1.625	1.75	2.5	1.67
PSA	2	1.3	1.7	2.125	2.25	3	2.00
PSX	3	.9	1.3	1.625	1.75	2.5	1.67
POCw&PSCw	1	.9	.8	1	1.5	1.25	.83
C	BLOAT		-.2	.2 align >A	0	.5 align >G	
CW	2	1.4	1.6	2	3	3.5	2.33
CS	2	1.8	1.6	2	3	2.5	1.67
CSG	2	1.8	1.8	2.25	3	3	2.00
DOC	0	1.2	1	1.25	2	1.5	1.00
XOC, XOCw	0	1.6	1.1	1.375	2.75	1.75	1.17
F	BLOAT		.8	-.6 align >C	1.5	-1.125 align >C	
FW	3	3.2	2.6	3.25	6	4.875	3.25
FS	3	1.6	2.2	2.75	3	4.125	2.75
Fp	6	4.8	4.8	6	9	9	6.00
FOC	1	1.3	.9	1.125	2.25	1.4375	.96
V	BLOAT		0	.4 align >F	0	.75 align >F	
VW	2	1.6	2	2.5	3	3.75	2.50
VS	2	1.6	1.2	1.5	3	2.25	1.50
FOV	1	1.2	.7	.875	2.25	1.3125	.88
VSC	2	1.7	1.4	1.75	3	2.375	1.58
GOV	2	1.5	1.2	1.5	3	2.375	1.58
VSG	2	1.7	1.6	2	3	2.875	1.92
DOV	1	1.9	1.6	2	3.5	2.875	1.92
VSD	3	1.3	1.2	1.5	2.5	2.375	1.58
VSA	2	1.3	1.5	1.875	2.25	2.625	1.75
AOV	2	1.9	1.3	1.625	3.75	2.625	1.75
V SX	3	.9	1.1	1.375	1.75	2.125	1.42
XOV	0	1.5	.9	1.125	2.75	1.625	1.08
S	BLOAT		.8	-.8 align >V	1.5	-1.5 align >V	
SW	3	3.2	2.4	3	6	4.5	3.00
SS	4	2.4	3.2	4	4.5	6	4.00
Sp	7	5.6	5.6	7	10.5	10.5	7.00
SOV	1	1.2	.6	.75	2.25	1.125	.75

2.5

DESIGN RULE CALCULATIONS  
FOR CMOS DESIGN RULES  
AT LAMBDA = .8 and 1.5 MICRONS

revision 1.2.5

VARIABLE LABEL	LAMBDA RULES	"1.2 micron" TECH				" 3 micron" TECH		
		ILAMBDA= as drawn	.8 microns mask dim microns	fin.dim microns	4 spots fin.dim lambda	ILAMBDA= mask dim microns	1.5 microns fin.dim microns	3 spots fin.dim lambda
W	BLOAT		0	0		-6	6	
WW		6	4.8	4.8	6	3	9	6.00
WS		6	4.8	4.8	6	15	9	6.00
A	BLOAT		.6	-.8 align >W		1.5	-1.5 align >W	
AW		2	2.2	1.4	1.75	4.5	3	2.00
AS		3	1.8	2.6	3.25	3	4.5	3.00
Ap		5	4	4	5	7.5	7.5	5.00
ApIWn		6	4.5	4.9	6.125	11.25	9	6.00
AnIWp		6	4.5	4.9	6.125	5.25	9	6.00
ApOWp		4	2.9	3.3	4.125	2.25	6	4.00
AnOWn		4	2.9	3.3	4.125	8.25	6	4.00
X	BLOAT		3	-.8 align >W		5.5	-1.5 align >W	
XS		6	1.8	2.6	3.25	3.5	5	3.33
XpIWn		7	4.1	4.5	5.625	10.75	8.5	5.67
XnIWp		7	4.1					
		4.5	5.625	4.75	8.5	5.67		
XpOWp		3	3.9	3.5	4.375	4.25	6.5	4.33
XnOWn		3	3.9	3.5	4.375	10.25	6.5	4.33
XSA		5	2.2	3	3.75	4	5.5	3.67
G	BLOAT		-.2	-.2 align >A		0	-.5 align >A	
GW		2	1.4	1.2	1.5	3	2.5	1.67
GS		2	1.8	2	2.5	3	3.5	2.33
Gp		4	3.2	3.2	4	6	6	4.00
GSD		4	2.2	2.4	3	4	4.5	3.00
GOA		2	1.2	1.5	1.875	2.25	2.75	1.83
AOG		2	2	1.7	2.125	3.75	3.25	2.17
GSA		1	.6	1.1	1.375	.75	1.75	1.17
GSX		3	1	1.5	1.875	1.75	2.75	1.83
GSXs		2	.2	.7	.875	.25	1.25	.83
D	BLOAT		2.2	-.2 align >A		4	-.5 align >A	
DS		5	1.8	2	2.5	3.5	4	2.67
DSA		3	1	1.5	1.875	1.75	2.75	1.83
DSAs		2	.2	.7	.875	.25	1.25	.83
DSX		4	.6	1.1	1.375	1.25	2.25	1.50

METAL2	S	bloat	1	-1 align >V
MIN.WIDTH	SW	3	4	3
MIN.SPACE	SS	4	3	4
MIN.PITCH	Sp	7	7	7
VIA OVERLAP	SOV	1	1.5	.75 .7

NOTE: >ALL EDGES of features must be defined on the lambda grid.

>If feature edges are separated by an odd number of half grids within a layer, the edges will "flop" in a random direction by 1/2 spot due to the round off in the pattern generation.

>There are two additional layers referenced in the design rule calculations on the following pages, both of which are transparent to the designer. The X layer is generated from a bloat of the active contact Ca and represents the active dogbone around the contact. This X layer is merged with the designer's A layer, after its bloat, to form the real ACTIVE mask layer. The D layer is generated from a bloat of the poly contact Cp and represents the poly dogbone around the contact. This D layer is merged with the designer's G layer, after its bloat, to form the real POLY1 mask layer. This procedure in the mask generation process facilitates an optimizing of the spacing between features for a particular technology and does not ordinarily concern the designer. The sizing of these features are calculated on the following pages.

J. Donald Trotter, Mississippi State University,  
(601)325-3751

NOTE: >C = Ca + Cp      The Ca and Cp "design" layers are the contacts to Active and Poly1, respectively. The Ca and Cp "design" layers merge on the mask to form the Contact mask layer. "C" without a subscript refers to both "design" layers.

CONTACT (ACTIVE) Ca	bloat		-.25	.25 align >A	
WIDTH CaW	2	1.75	2		
MIN.SPACE CaS	2	2.25	2		
ACTIVE TO A.CONTACT ASCa	5	4.75	5.125		
A.CONT TO POLY GATE CaSG	2	2.25	2.25		1
ACTIVE OVERLAP AOCa	1	2	1.375		.7
FIELD POLY TO A.CONT. GfSCa	3	3.25	3.25		1
above for < 5 lambda GfSCas	2	2.25	2.25		1

NOTE: >GfSCas and CgSAs allow one contact only.

CONTACT (POLY1) Cp	bloat		-.25	.25 align >A	
WIDTH CpW	2	1.75	2		
MIN.SPACE CpS	2	2.25	2		
POLY CONT TO ACTIVE CpSA	3	2.75	3.125		.7
above for < 5 lambda CpSAs	2	1.75	2.125		.7
P.CONT TO A.CONT CpSCa	4	4.25	4		

POLY1 OVERLAP GOCp	1	1.5	1.25		1
SPACE TO P.CONTACT GSCp	4	4.25	4.25		1

NOTE: >Both N-Well and P-Well contacts are required.  
 >The shorting contact is not allowed.  
 >Only the 2x2 (lambda) contact is allowed.

METAL1 F	bloat		1	-.75 align >C	
MIN.WIDTH FW	3	4	3.25		
MIN.SPACE FS	3	2	2.75		
MIN.PITCH Fp	6	6	6		
CONTACT OVERLAP FOC	1	1.625	1.125		.7

VIA V	bloat		0	.5 align >F	
VIA WIDTH VW	2	2	2.5		
VIA SPACE VS	2	2	1.5		
1st METAL OVERLAP FOV	1	1.5	.875		.7
VIA SPACE TO CONT. VSC	2	2.125	1.75		1
GATE OVERLAP (VIA) GOV	2	1.875	1.5		1.4
SPACE TO POLY GATE VSG	2	2.125	2		1.4
VIA TO ACTIVE VSA	2	1.625	1.875		1.2
ACTIVE OVERLAP (VIA) AOV	2	2.375	1.625		1.2

NOTE: >Vias over active area are permitted.  
 >Only a via size of (2x2) is allowed.  
 >The stacked contact (a via stacked over a contact) is not allowed.



## 2.4 CMOS Design Rules for Lambda-Based Designs

These lambda-based design rules facilitate designing for the 1.2 micron technology and processing at either 1.2 or 3 microns. The lambda dimensions are for convenience and do not correspond to mask or finished dimensions.

Bloat for each layer refers to the expansion in total linewidth, going from the drawn dimension to the mask dimension, and then to the finished dimension on the finished wafer. As an example, the minimum active line width is drawn as 2 lambda. It is bloated to 2.75 lambda at the mask and then shrinks 1 lambda in wafer processing to a finished wafer linewidth of 1.75 lambda.

DESCRIPTION	VARIABLE LABEL		LAMBDA Drawn	RULES Mask	Finished	Assumed Misalignment
WELL	W	bloat		0	0	
WELL WIDTH	WW		6	6	6	
WELL SPACING	WS		6	6	6	
ACTIVE TRANSISTOR	A	bloat		.75	-1 align	>W
MIN.WIDTH	AW		2	2.75	1.75	
MIN.SPACE	AS		3	2.25	3.25	
MIN.PITCH	Ap		5	5	5	
P+INSIDE N WELL	ApiWn		6	5.625	6.125	.7
N+INSIDE P WELL	AniWp		6	5.625	6.125	.7
P+ Overlap P WELL	ApOWp		4	3.625	4.125	.7
N+ Overlap N WELL	AnOWn		4	3.625	4.125	.7
GATE	G	bloat		-.25	-.25 align	>A
MIN.WIDTH	GW		2	1.75	1.5	
MIN.SPACE	GS		2	2.25	2.5	
MIN.PITCH	Gp		4	4	4	
GATE OVERLAP	GOA		2	1.5	1.875	.7
ACTIVE O. of GATE	AOG		2	2.5	2.125	.7
POLY TO ACTIVE	GSA		1	.75	1.375	.7
P+	P	bloat		0	0 align	>A
SPACE TO Op.Gate	PSG & POG		3	3.125	3.25	1
OVERLAP OF ACTIVE	POA		2	1.625	2.125	.7
SPACE TO Op.Active	PSA		2	1.625	2.125	.7
WELL CONT.OVERLAP	POCw&PSCw		1	1.125	1	1

reproduction purposes. These calculations were performed using a spreadsheet to facilitate the what-if nature of the exercise. Emphasis is on the 1.2 microns technology in conjunction with the 3 microns; however, two additional levels of scaling are purposed for consideration by MOSIS: a more aggressive 1.2 microns, corresponding to VHSIC dimensions (roughly), and 2 microns (or 2.4 microns, depending on your definition), corresponding to the believed present lower limit for full wafer exposure.

The rules are intended to support N-well, P-well, and twin-tub processing, although the exact interface to MOSIS for the N-well is still to be determined in the next few months. A design to accommodate all varieties must have appropriate well (substrate) contacts for both polarities. A well around the die with the same polarity as the substrate is required to prevent junction leakage at the edge of the chip. It is still to be determined if MOSIS or the designer is expected to provide this feature. The nominal drawn P-well is drawn equal distance between the NMOS and PMOS transistors. Appropriate bloating and/or shrinking is to be provided by MOSIS.

base can control the flop to always be in a consistent direction by disallowing the exact  $1/2$  spot location.

Consider a very important example. In the Mead and Conway rules, the active layer overlap of the contact mask is drawn as one lambda. However, as the technologies scale down, the alignment errors of the alignment jigs become a more significant portion of this overlap tolerance. For fear of yield losses due to the metal shorting to the substrate by virtue of the metal contacting the substrate, some vendors have increased the requirement for the active overlap of the contact on a relative basis as the technology is scaled down. How should the designer draw the overlap? 1.5 lambda maybe, since that corresponds to an acceptable increase! That half lambda edge will not result in an uncontrolled flop of the feature at the 1.2 microns since there are two drawing units in the half lambda. However, at the 3 micron technology, the edge falls in the middle of the spot. It can be controlled to always flop in the positive direction or vice versa, however, what is needed is for the feature edge to move relative to the position of the contact. How is this to be achieved?

Bloating of the overlap structures around the contacts is a key requirement in providing the flexibility for scaling. Should MOSIS bloat, as an example, the entire active region or just the active around the contact? How should MOSIS handle the poly layer since the gate length needs to be independently controlled versus the poly overlap of the contact (poly dogbone)?

After considerable effort, discussions with MOSIS programmers, and discussions with designers in the community, the collective decision was made to draw for convenience with an active overlap and poly overlap of one lambda. In addition, the designer is asked to distinguish the poly contact from the active contact in order for MOSIS to more easily individually bloat the four structures: active and transistors, active contacts, poly and poly gates, and poly contacts. As an example, in order for MOSIS to generate the active mask definition for the pattern generator, MOSIS will bloat all the active contacts an appropriate amount and merge these features with the separately bloated active features. The poly layer is treated similarly. Although MOSIS could extract the distinction of the contacts from logic operations on the data base, the extra computing cost redirects the solution to the designer maintaining that information with the design submission.

With the philosophy worked out, what remains is the determination of the MOSIS bloats and shrinks on each layer and the individual design rules. The next two sections contain this information in a rather compact form for

Without proof, the following observation is provided: Twice lambda minus one drawing unit should equal the minimum resolution of the lithography. This underscores the ability to increase (bloat) the size of certain neighboring features by one drawing unit per edge without violating the minimum resolution requirement.

These observations with lots of detailed layouts and playing the "what-if" game have led to the following selections (in microns):

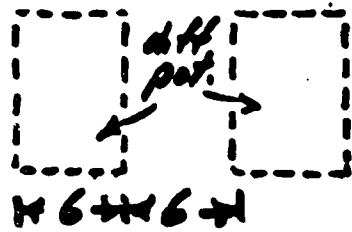
	technology	
	1.2 microns	3 microns
lambda	0.8	1.5
drawing unit	0.2	0.5
spots/lambda	4	3
minimum resolution	1.4	2.5

These arguments have led to the basis for the feature sizing desired on the mask and which support rather straight forward extension of the original Mead and Conway representation of device and interconnect features. However, there is the question: based on the provided lambda design, by what algorithms are the features sized by MOSIS ?

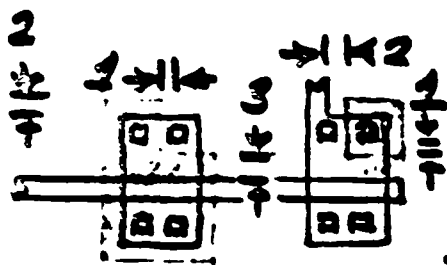
There are at least six different design methodologies used in the community to develop and define the mask artwork information. Center defined features provide the most simple extension to bloating (or shrinking) the features required by MOSIS; however, MAGIC and CAESAR are two edge defined systems essential to the community. A solution is necessary to accommodate these systems.

A little detail that is important in the MOSIS system is that there cannot be an absolute reference dependency in sizing the features since MOSIS relocates designs independently of the lambda grid. What happens when the edge definition of a feature does not fall on the edge of the raster for the e-beam? The software rounds the feature edge off to the nearest raster edge. In other words, the feature "flops" to the nearest raster. If the feature size is an integer number of spot sizes (drawing units), the pattern will flop toward the smallest distance but still maintain the feature size as drawn. If the feature edge falls directly in the middle of a spot, the flop will be in a random direction with half of a drawing unit displacement. Providing a very small bias to all the numbers in the data

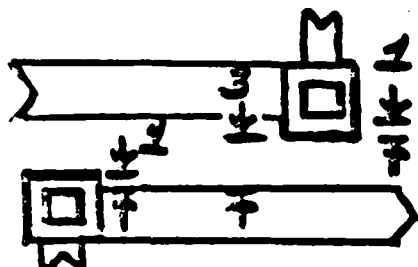
## SCALABLE CMES RULES



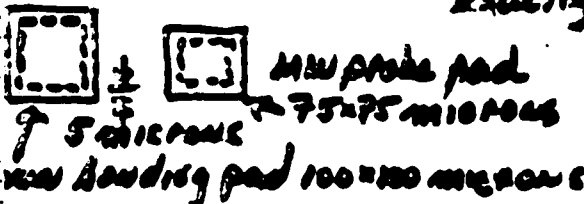
**PSELECT  
(NSELECT)**



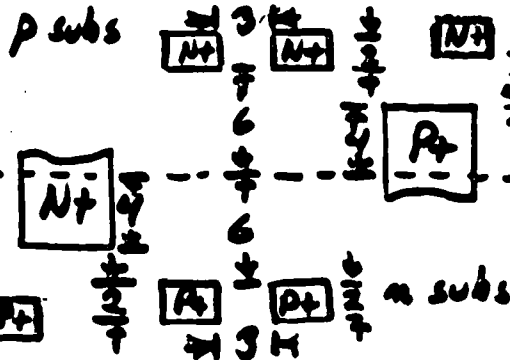
# METAL 1



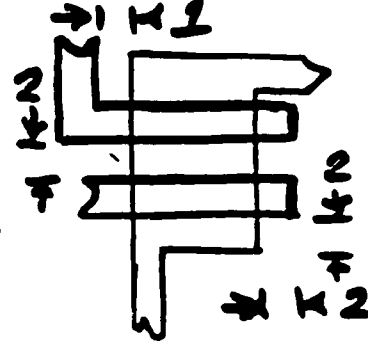
# GLASS



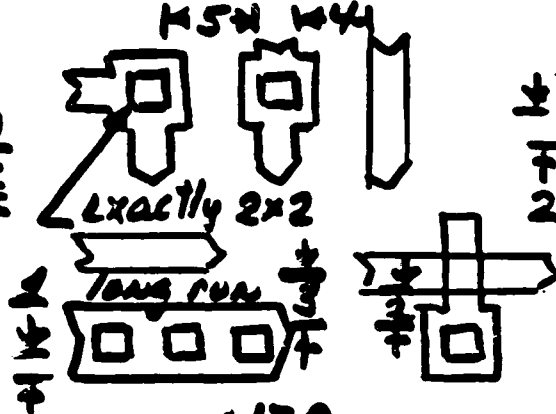
# ACTIVE



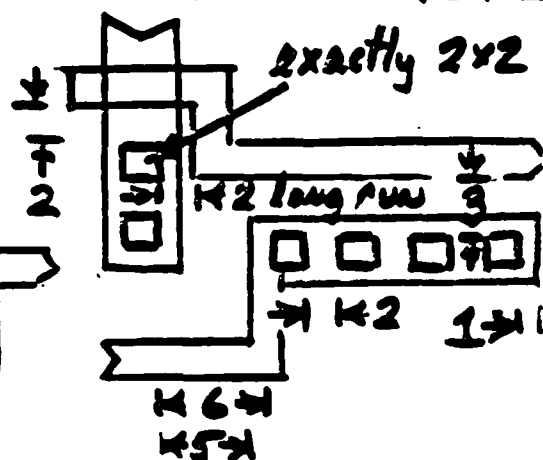
POLY



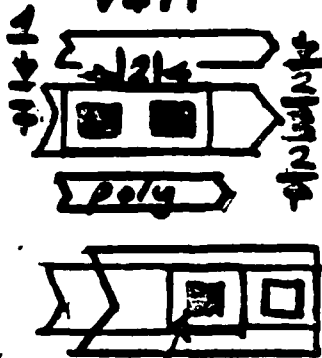
## CONTACT TO POLY



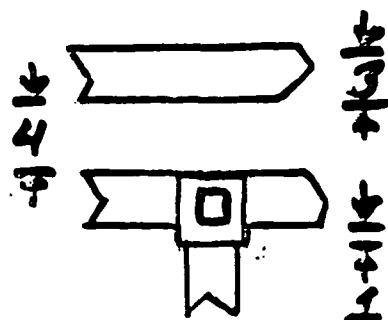
## CONTACT TO ACTIVE



VIA



# METAL 2



LYR NAMES, COLORS, C/F EXT:

NSELECT CSN METALY CMF

POLY CPG VIA CVA

Cont. to A4y CCP METAL2 CHS


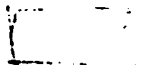
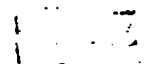




Cont. to Act. CCA CLASS COG

P WELL  
A WELL  
ACTIVE  
P SELECT

CWP  
CWN  
CAA  
CSP

NSELECT	CSN	METAL1	CMF
POLY	CPG	VIA	CVA
Cont. to Aly	CCP	METAL2	CHS
Cont. to Act.	CCA	GLASS	CGG

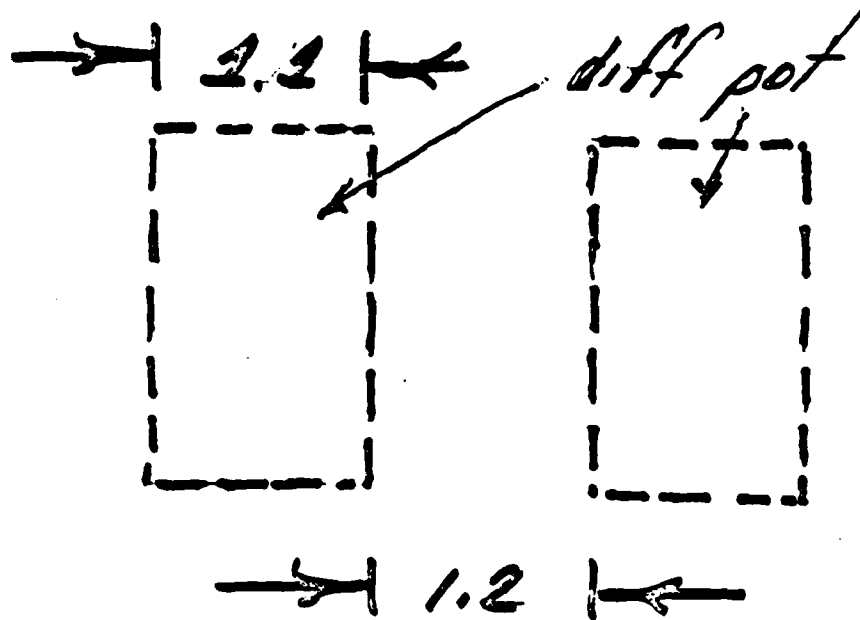
# LAYER NAMES, EXTENSIONS, AND COLORS

NAME	CIF EXT	CALMA#	COL
PWELL	CWP	41	BROWN
NWELL	CWN	42	RED
ACTIVE	CAA	43	
PSELECT	CSP	44	
NSELECT	C SN	45	
POLY	EPG	46	
CONTACT TO POLY	CCP	47	BLACK
CONTACT TO ACTIVE	CCA	48	BLACK
METAL 1	CMF	49	
VIA	CVA	50	
METAL 2	CMS	51	
GLASS	COG	52	BLACK

# 1. PWELL (NWELL)

1.1 WELL WIDTH  $\lambda$  6

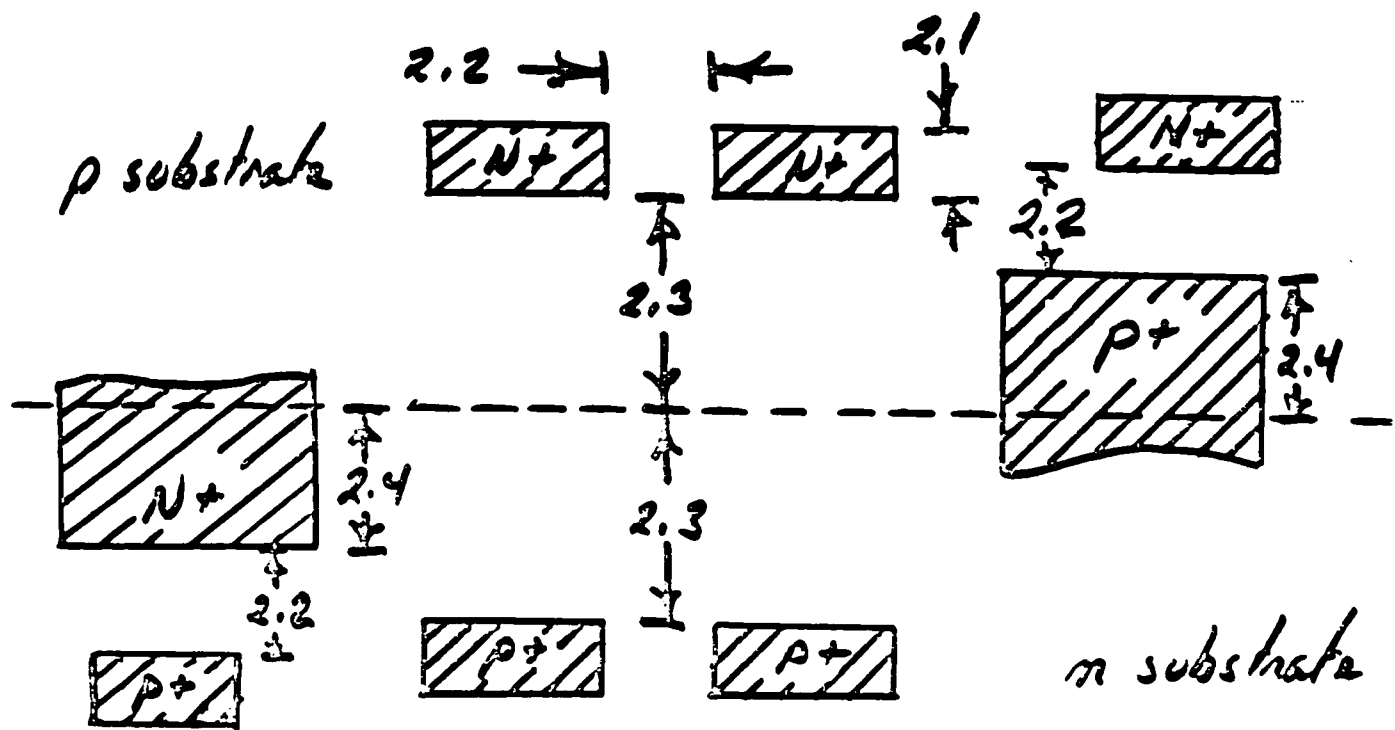
1.2 WELL SPACE 6



Note: if both p and n well layers submitted these may not overlap. They may be coincident.

## 2. ACTIVE

	Lambda
2.1 MIN width	2
2.2 MIN space	3
2.3 Source/drain active to well edge	6
2.4 Substrate contact active overlap of substrate (for guard bars + guard rings)	4

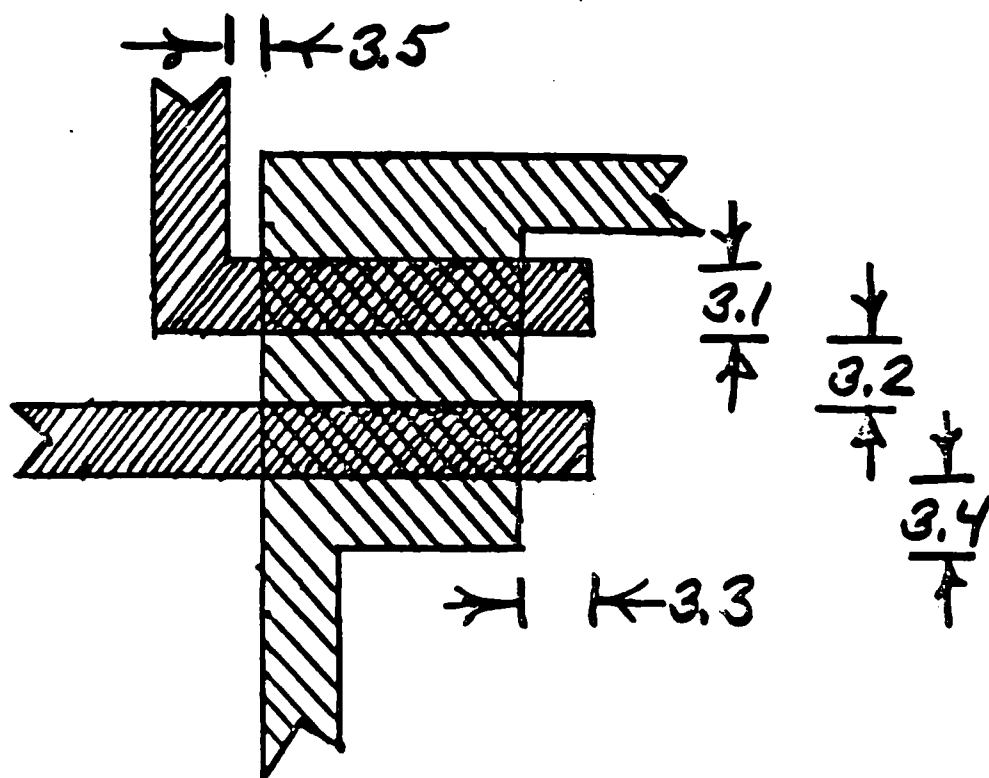




# 3. POLY

Lambda

3.1	MIN width	2
3.2	MIN space	2
3.3	Gate overlap of active	2
3.4	Active overlap of gate	2
3.5	Field poly to active	1



# 4. P SELECT (N SELECT)

Lambda

4.1 P select space (overlap)  
to (of) gate

3

4.2 P select space (overlap)  
to (of) active

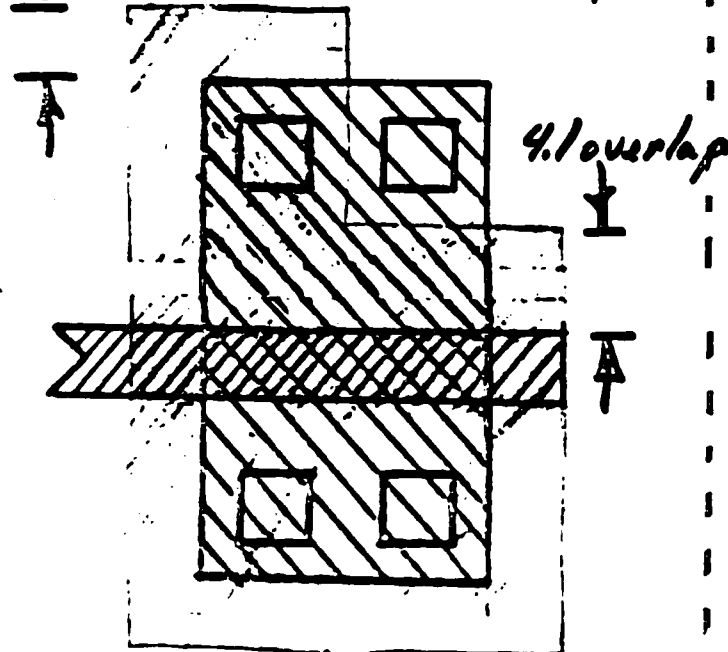
2

4.3 P select space (overlap)  
to (of) contact to subs.

1

4.2 overlap

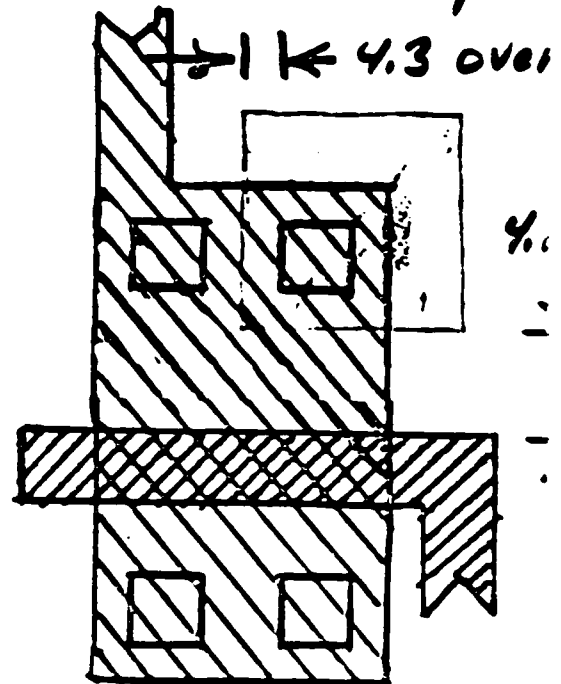
4.3 space



out of p well

4.2 spac

4.3 over

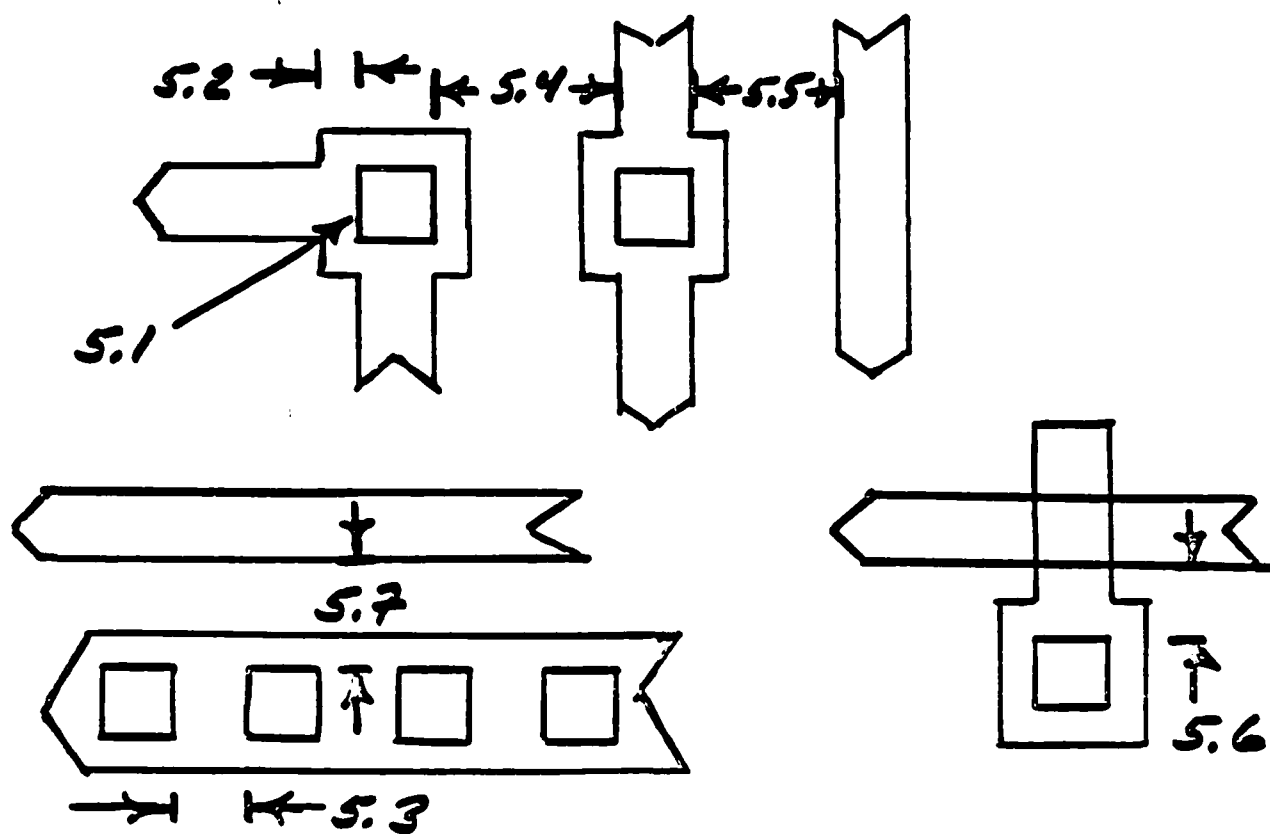


in p well

Note: If both p select & n select layers submitted they may be coincident. They must not be

## 5. CONTACT TO POLY \*

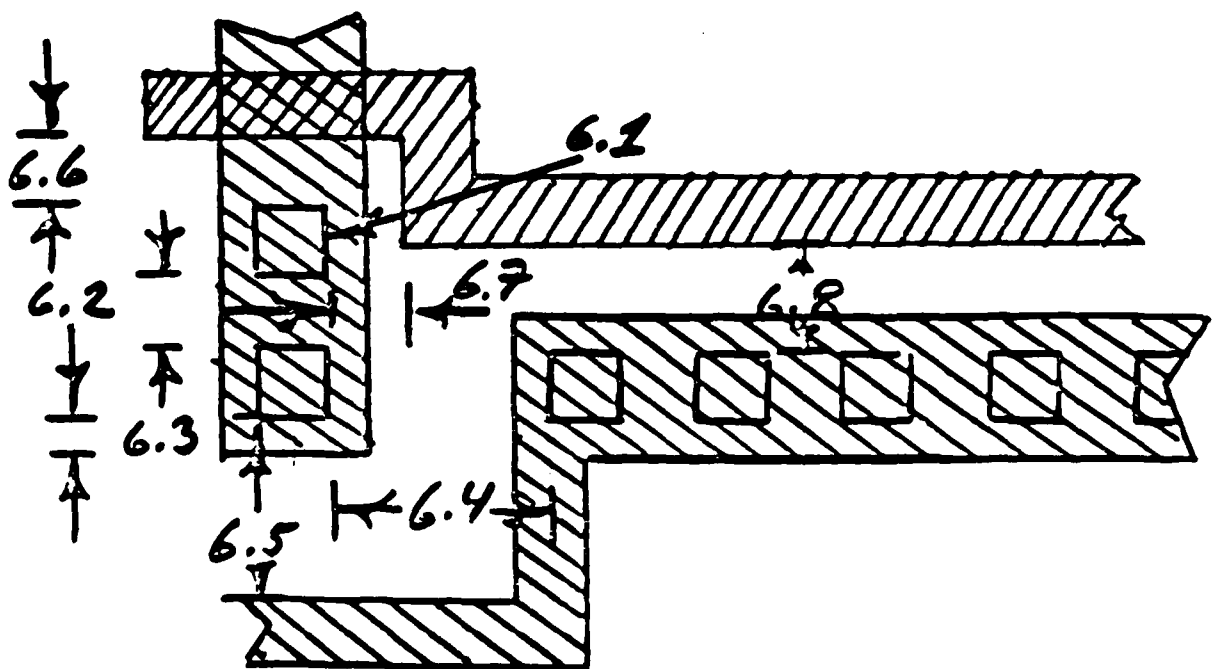
	Lambda
5.1 Contact size, exactly	2x2
5.2 Poly overlap of contact	1
5.3 Spacing on same poly	2
5.4 Spacing on different poly	5
5.5 Space to other poly	4
5.6 Space to active, short run	2
5.7 Space to active, long run	3



\* Your associating contacts with the poly or active layer allows ADSIS to independently float the layer and the layer overlap of the contact.

## 6. CONTACT TO ACTIVE

		Units
6.1	Contact size, exactly	2x2
6.2	Active overlap	1
6.3	Spacing on same active	2
6.4	Spacing on different active	6
6.5	Space to different active	5
6.6	Space to gate	2
6.7	Space to poly over field, short run	2
6.8	Space to poly over field, long run (to minimize ringing capacitance)	3



## 7. METAL 1

Lambda

7.1 MIN width

3

7.2 MIN space to metal 1

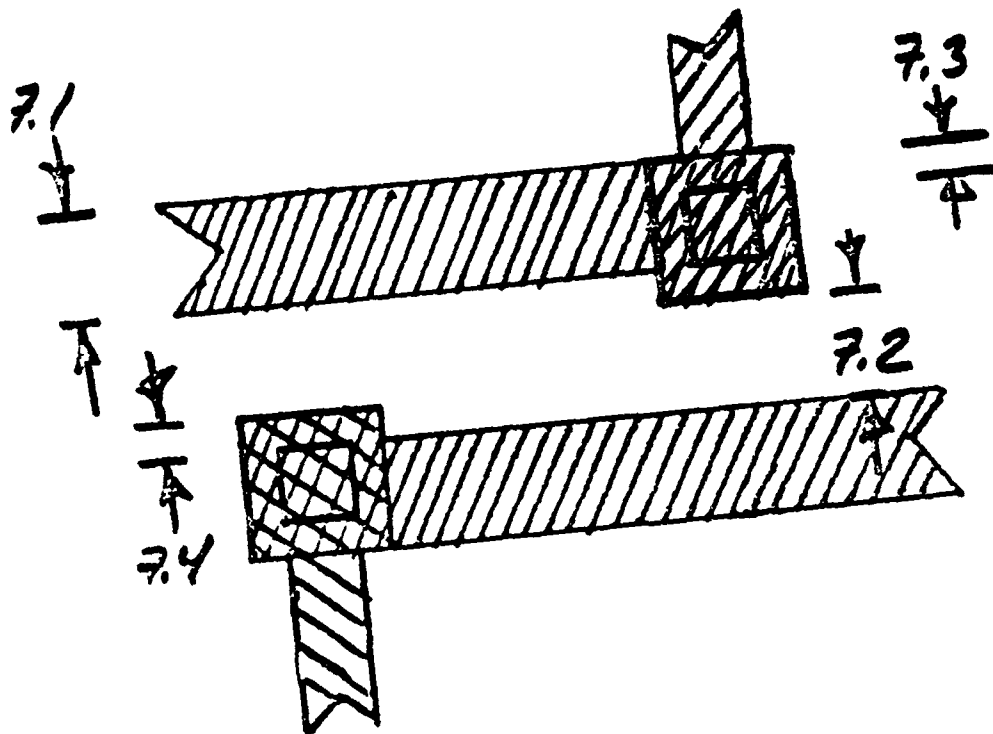
3

7.3 MIN poly contact overlap

1

7.4 MIN active contact overlap

1

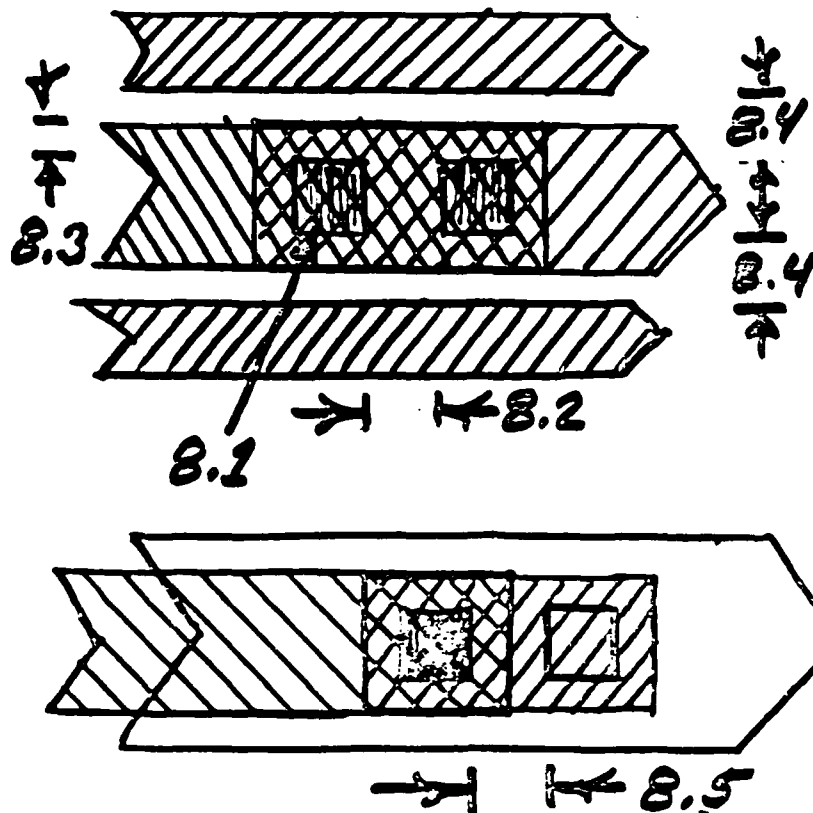


## 8. VIA

Lambda

8.1	via size, exactly	2x2
8.2	separation	2
8.3	metal 2 overlap	1
8.4	space to poly or active edge	2
8.5	space to contact	2

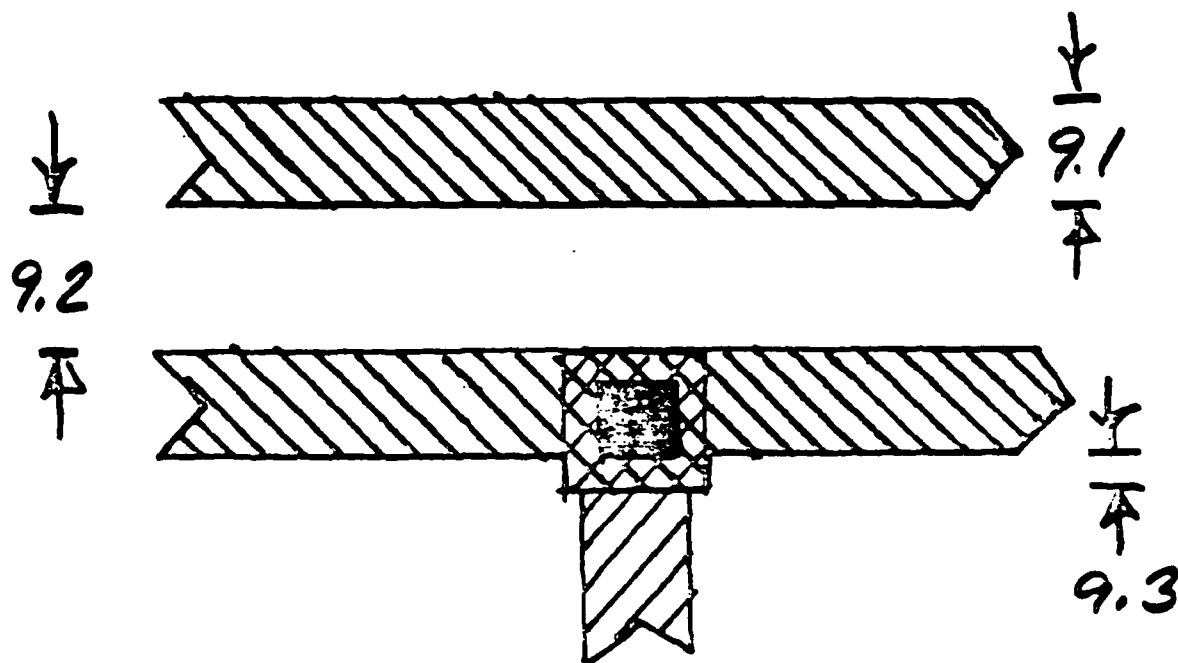
objective:  
keep via on  
a flat  
surface



Note: via stacked over a contact not allowed

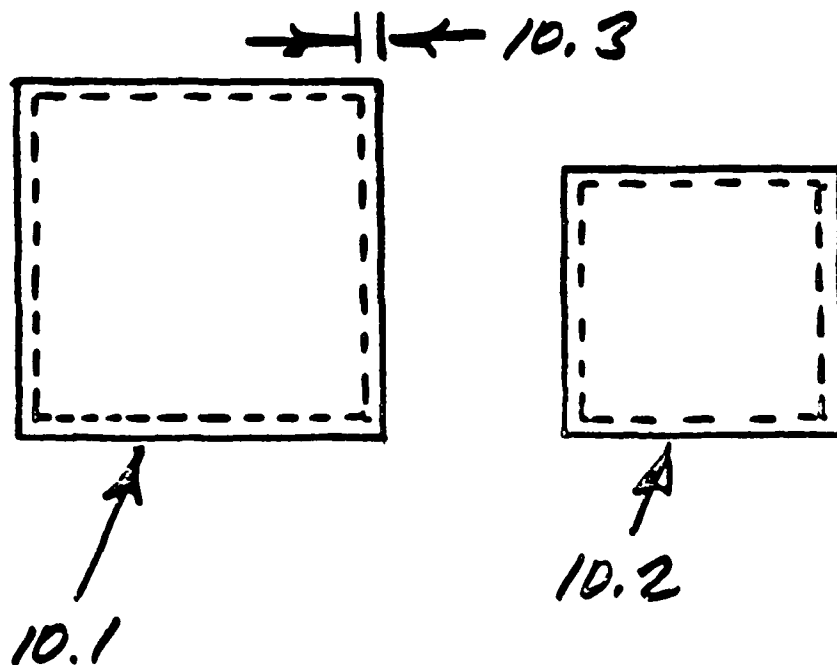
## 9. METAL 2

	Lambda
9.1 Min width	3
9.2 Min space	4
9.3 Min via overlap	1



## 10. GLASS

	microns
10.1 Min bonding pad	100 x 10
10.2 Min probe pad	75 x 75
10.3 Pad to glass edge	5





# TECHNOLOGY NAMES AND REQUIRED LAYERS

Process	Technology name
---------	-----------------

P WELL	SCP
included must be layers CWP, CSP	

N WELL	SCAN
included must be layers CWN, CSN	

TWIN TUB p substrate	SCTP
included must be layers CWP, CWN, CSP, or CWN, CSN	

TWIN TUB n substrate	SCTN
included must be layers CWP, CWN, CSP, C or CWP, CSP	

WHAT VALUES  
LAMBDA?

LAMBDA = 1.5 MICRONS  
for current 3 micron  
fabricators

LAMBDA = 0.8 MICRONS  
for current 1.2 micron  
fabricators

interconnections and transistors. However, clocked CMOS, with the concept of precharging circuit nodes and then conditionally discharging them, requires only one transistor per signal input [3]. PLA design with clocked CMOS has been developed with this distinct advantage in mind.

The Path Programmable Logic (PPL) is a variation of a PLA in which the AND array and the OR array are folded together so that input lines and output lines are alternated within a single array. Column and row breaks in the AND and OR planes allow the design of independent arrays on the same integrated circuit. This concept, which is familiar as Storage/Logic Array (SLA), was originally published by Suhas Patil [1]. This project also explores the possibility of implementing a PPL with CMOS and compares it with other PLA implementations.

The process used for fabrication is a silicon-gate P-well process with double layer metal. The lambda based scalable design rules facilitate designing for 1.2 micron and then processing at either 1.2 or 3 microns. The legend of the layers used and the documentation of the design rules are included in Appendix B.

plane. The AND plane generates specific logic combinations of the inputs (implicants) which are run through the OR plane. The outputs of the OR plane are stored in output latches during phase 2. PLA implementations vary for different technologies.

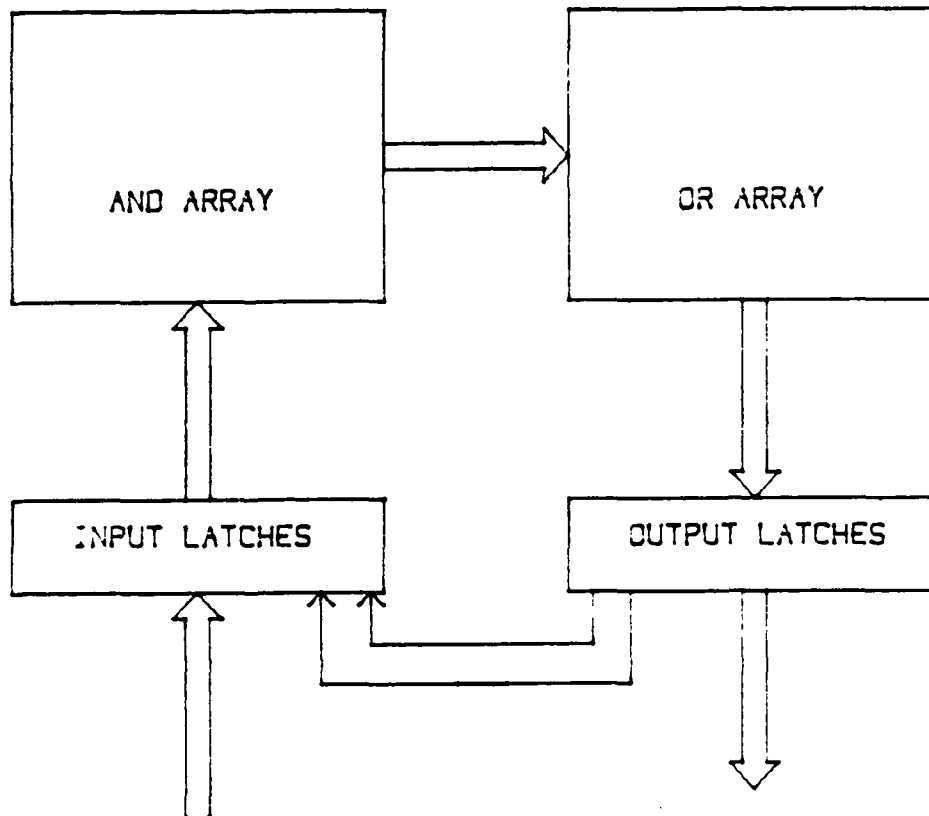


Figure 1.1 General PLA Block Diagram

This project explores PLA design using CMOS technology. Classic CMOS circuitry requires signals to drive pairs of transistors instead of one, as is normally used in NMOS. This requires greater area for

## CHAPTER I

### INTRODUCTION

The advent of Very Large Scale Integrated (VLSI) circuit technology illustrates the continuing trend toward increased gate counts on logic chips. Because of the complexity and cost of designing such chips, a structured form of logic implementation is desirable [1]. A Programmable Logic Array (PLA) is an orderly structure for handling combinational sequential logic functions [2]. PLA's have become increasingly popular as a means of implementing complex logic with reduced engineering costs [3]. The PLA implementation has the advantage of giving designers more flexibility because logic changes can be made easily and quickly. PLA's can realize arbitrary logical functions and can be used in microprogramming controllers as well as in random logic circuits.

The PLA, which is conceptually a two-level AND-OR, is attractive in LSI because of its memory-like regular structure [4]. In contrast with ROM, only implicants and not minterms of boolean functions are stored, so that it is not necessary to have a word of storage for every input combination. The overall structure of a PLA is illustrated in Figure 1.1. The inputs provided with temporary storage during phase 1 by input latches are run through the AND

LIST OF TABLES

Table		Page
3.1	16 by 4 priority encoder. . . . .	40
3.2	Seven segment display . . . . .	40
3.3	Data for input partitioning . . . . .	41
3.4	Table of inputs to the PLA. . . . .	43
3.5	Table of outputs from the PLA . . . . .	44
4.1	Controller for subroutine stack and multiplexer of microsequencer . . . . .	62
4.2	Moderate PLA example. . . . .	65
4.3	Traffic light controller. . . . .	70
5.1	Excitation table for TLC. . . . .	89
5.2	Modified excitation table for TLC . . . . .	90
5.3	Minimized table derived from 5.2. . . . .	91
6.1	Comparison of AND-OR planes of different PLA's . . . . .	104
6.2	Circuit area for 4-inputs & 3-outputs using different PLA's . . . . .	104

5.4	Input latch for PPL . . . . .	78
5.5	NMOSTLY latch . . . . .	79
5.6	Set-Reset latch . . . . .	80
5.7	Timing diagram of PPL . . . . .	80
5.8	Floor plan of AND & OR cells. . . . .	82
5.9	Symbolic representation of Exclusive-OR . . . .	86
5.10	AND-OR functioning. . . . .	87
5.11	Symbolic program for TLC. . . . .	92
5.12	Minimized symbolic program of TLC . . . . .	93
5.13	Work sheet for chromatics . . . . .	95
5.14	Layout of the TLC . . . . .	96
5.15	Pin assignments of the PPL chip . . . . .	97
5.16	The PPL chip. . . . .	98

3.21	Layout of Set-Reset latch . . . . .	38
3.22	Layout of input latches for Large PLA . . . . .	45
3.23	Layout of output latches for Large PLA. . . . .	46
3.24	Layout of the AND plane for Large PLA . . . . .	47
3.25	Layout of the OR plane for Large PLA. . . . .	48
3.26	Pin assignments for the Large PLA chip. . . . .	49
3.27	The Large PLA chip. . . . .	50
4.1	AND-OR of moderate PLA. . . . .	51
4.2	AND-OR circuit for moderate PLA . . . . .	53
4.3	Layout of AND plane for 2 implicants. . . . .	54
4.4	Layout of OR configuration for moderate PLA . . . . .	55
4.5	Gated input circuitry . . . . .	56
4.6	Input latch for moderate PLA. . . . .	57
4.7	Dynamic NMOSTLY latch . . . . .	58
4.8	Output latch for moderate PLA . . . . .	59
4.9	Overall circuit of moderate PLA . . . . .	60
4.10	Timing diagram of moderate PLA. . . . .	61
4.11	Layout of the AND plane for moderate PLA. . . . .	63
4.12	Layout of the OR plane for moderate PLA . . . . .	64
4.13	Circuit schematic of small ripple PLA . . . . .	67
4.14	Timing diagram of small PLA . . . . .	68
4.15	Layout of the small PLA . . . . .	69
4.16	Pin assignments of the chip . . . . .	71
4.17	The final chip drawing. . . . .	72
5.1	NOR-NOR implementation. . . . .	75
5.2	NAND-NAND implementation. . . . .	76
5.3	Improvised circuit for AND-OR to avoid change splitting . . . . .	77



LIST OF ILLUSTRATIONS

Figure		Page
1.1	General PLA block diagram . . . . .	2
2.1	Clocked CMOS circuits . . . . .	5
2.2	Ripple CMOS circuits. . . . .	8
2.3	Gated CMOS circuits . . . . .	9
3.1	Possible combinations for the OR plane. . . . .	12
3.2	Possible combinations for the AND plane . . . . .	13
3.3	Dynamic CMOS PLA with input partitioning. . . . .	14
3.4	Timing diagram of gated PLA . . . . .	15
3.5	Schematic of input latch. . . . .	16
3.6	CMOS transmission gate. . . . .	17
3.7	Input latch with partitioning . . . . .	19
3.8	Input partitioning cells. . . . .	21
3.9	Layout of input circuit . . . . .	22
3.10	Layout of circuit with input partitioning . . . . .	23
3.11	Layout of circuit without input partitioning. . . . .	24
3.12	AND-OR circuit. . . . .	25
3.13	Layout of AND-OR configuration for Large PLA. . . . .	26
3.14	AND-OR layout with active areas programmed. . . . .	28
3.15	Diffusion cells programmed into AND-OR planes . . . . .	30
3.16	Delay signal circuit schematic. . . . .	31
3.17	Layout of delay signal circuit. . . . .	33
3.18	Pmostly latch . . . . .	34
3.19	Layout of dynamic Pmostly latch . . . . .	36
3.20	Set-Reset latch . . . . .	37

	Programming . . . . .	39
CHAPTER 4	<u>RIPPLE PLA'S</u> . . . . .	51
4.1	Moderate PLA . . . . .	51
	The Moderate PLA design example . . . . .	61
4.2	Small PLA . . . . .	65
	Circuit description. . . . .	66
	The Design example . . . . .	66
	Chip frame . . . . .	68
CHAPTER 5	<u>PATH PROGRAMMABLE LOGIC</u> . . . . .	73
5.1	Introduction. . . . .	73
5.2	PPL circuit . . . . .	73
5.3	PPL cell set . . . . .	81
5.4	Symbolic representation . . . . .	85
5.5	Design methodology. . . . .	86
5.6	PPL example . . . . .	89
CHAPTER 6	<u>CONCLUSIONS</u> . . . . .	99
APPENDIX A	<u>PASCAL PROGRAM</u> . . . . .	105
APPENDIX B	<u>SCALABLE DESIGN RULES</u> . . . . .	115
APPENDIX C	<u>PLA'S CELL LIBRARY</u> . . . . .	131
APPENDIX D	<u>PPL CELL SET</u> . . . . .	159
	BIBLIOGRAPHY . . . . .	178

# TABLE OF CONTENTS

Preface . . . . .	iii
Abstract . . . . .	v
List of Illustrations . . . . .	viii
List of Tables. . . . .	xi
 CHAPTER 1 <u>INTRODUCTION</u> . . . . .	 1
 CHAPTER 2 <u>BASIC CMOS CIRCUIT CONCEPTS.</u> . . . . .	 4
2.1    Clocked CMOS circuits. . . . .	4
2.2    Types of CMOS gates. . . . .	7
Ripple circuits . . . . .	7
Gated circuits. . . . .	10
 CHAPTER 3 <u>LARGE PLA</u> . . . . .	 12
3.1    PLA with parallel gates. . . . .	12
3.2    Typical cells. . . . .	16
Input circuit . . . . .	16
AND-OR. . . . .	20
Programming the PLA. . . . .	27
Delay circuit . . . . .	29
Pmostly latch . . . . .	32
Set-Reset latch . . . . .	35
3.3    The PLA design example . . . . .	37
Priority encoder. . . . .	37
Seven segment display . . . . .	39
Test signals . . . . .	39

v

## ABSTRACT

Ajay Kumar Reddy Naini, Master of Science, June, 1985

Major : Electrical Engineering,  
Department of Electrical Engineering

Title of Thesis: CMOS Approaches to PLA Designs

Directed by : J. Donald Trotter

Pages in Thesis: 189      Words in Abstract: 119

## ABSTRACT

Various Programmable Logic Array (PLA) implementations with clocked CMOS technology are explored in this project. Three different CMOS PLA circuit styles are described: the "large" PLA uses a gated OR plane and is useful for a system with large number of inputs; the "moderate" PLA and the "small" PLA are ripple varieties with the former having the capability of handling a larger number of inputs than the latter. Path Programmable Logic (PPL), which is a folded form of a PLA, is also studied. A symbolic form of representation is developed and future PPL development activities are discussed. The PPL approach has a size and flexibility advantage over the other PLA approaches - except in applications requiring large PLA's.

circuit schematics and layouts. Chapter VI discusses in detail the advantages and disadvantages of all the PLA's designed, relative to their application, circuit area and performance.

I wish to express sincere thanks and gratitude to my major Professor, Dr. J. Donald Trotter, whose enthusiasm, continued guidance, and assistance made this thesis possible. My sincere thanks are due as well to Dr. William A. Hornfeck for serving on my committee. The final manuscript also has benefitted from the careful reading and comments of Dr. Howard D. Embree. I would also like to thank my friends, my fellow students, and the staff in the Microelectronics-Design laboratory for their timely assistance during the course of this work. My thanks are due to Mr. Ravi Nalavade for his help and assistance on this work.

Finally, I would like to thank my parents for their encouragement and personal sacrifices which enabled me to pursue higher studies and accomplish this research.

## PREFACE

This project explores custom CMOS design methodologies for different Programmable Logic Array (PLA) implementations often used in the circuitry which controls the functional blocks of Microcomputers.

Chapter I reviews the basic theory of PLA implementation and discusses CMOS designs and the fabrication process. Chapter II explains clocked-CMOS circuit concepts, both ripple and gated versions. Chapter III describes a PLA design using a parallel-transistor configuration suitable for a large number of inputs. To program this PLA, a pascal program is developed and documented in Appendix A. Chapter IV describes ripple PLA designs suitable for a small number of inputs/implicants. The "small" PLA version described can accommodate a smaller number of implicants, but it is faster than the other PLA's described. The "moderate" size PLA is designed to accommodate a larger number of inputs at the cost of additional circuitry. Chapter V discusses Path Programmable Logic (PPL); a PPL cell set is defined and a design approach using symbolic representation is developed. In Appendix B lambda-based scalable design rules used for the above designs are documented, and exceptions to the design rules are discussed. The various cells of all PLA's discussed are provided in Appendix C, which contains

CMOS APPROACHES TO PLA DESIGNS

BY

AJAY KUMAR REDDY NAINI

A Thesis  
Submitted to the Faculty of  
Mississippi State University  
in Partial Fulfillment of the Requirements  
for the Degree of Master of Science  
in the Department of Electrical Engineering

Mississippi State, Mississippi

July 1985

# INSTRUCTIONS

DRAW IN LAMBDA AND  
STAY ON A LAMBDA  
GRID

SCALE YOUR DESIGN TO  
CENTIMICRONS. YOUR  
CIF SHOULD NEVER  
REPRESENT CENTILAMBDA

WHAT YOU DRAW WILL BE  
CLOSE TO WHAT YOU GET  
ON Si. PROSIS WILL TELL  
YOU THE DIFFERENCES



## CHAPTER II

BASIC CMOS CIRCUIT CONCEPTS2.1 Clocked CMOS Circuits

Clocked CMOS circuitry is based on the concept of precharged gates. A three input NOR gate can be implemented with three NMOS transistors in parallel and a single PMOS transistor as load or precharge device (Figure 2.1). The PMOS transistor precharges the output node unconditionally high (logic "1") when phase1 goes high. The logic is evaluated by the NMOS transistors which conditionally discharge the output node to ground. It should be noted that the precharge device should be OFF when the logic is being evaluated. Therefore logic is evaluated when phase1' is high. As the NMOS transistors are in parallel, any one of the inputs going high discharges the output node, hence the NOR function. If the evaluation logic does not offer a path to ground, the output node remains in the precharged state. The output node with an associated leakage time constant is dynamic in nature and should be sampled with appropriate timing constraints. This type of circuit will be referred to as N-gate. The NMOS devices arranged in series provide the NAND function. A more general analysis of the circuit can be done in a similar fashion.

The use of an evaluation device (Figure 2.3) in

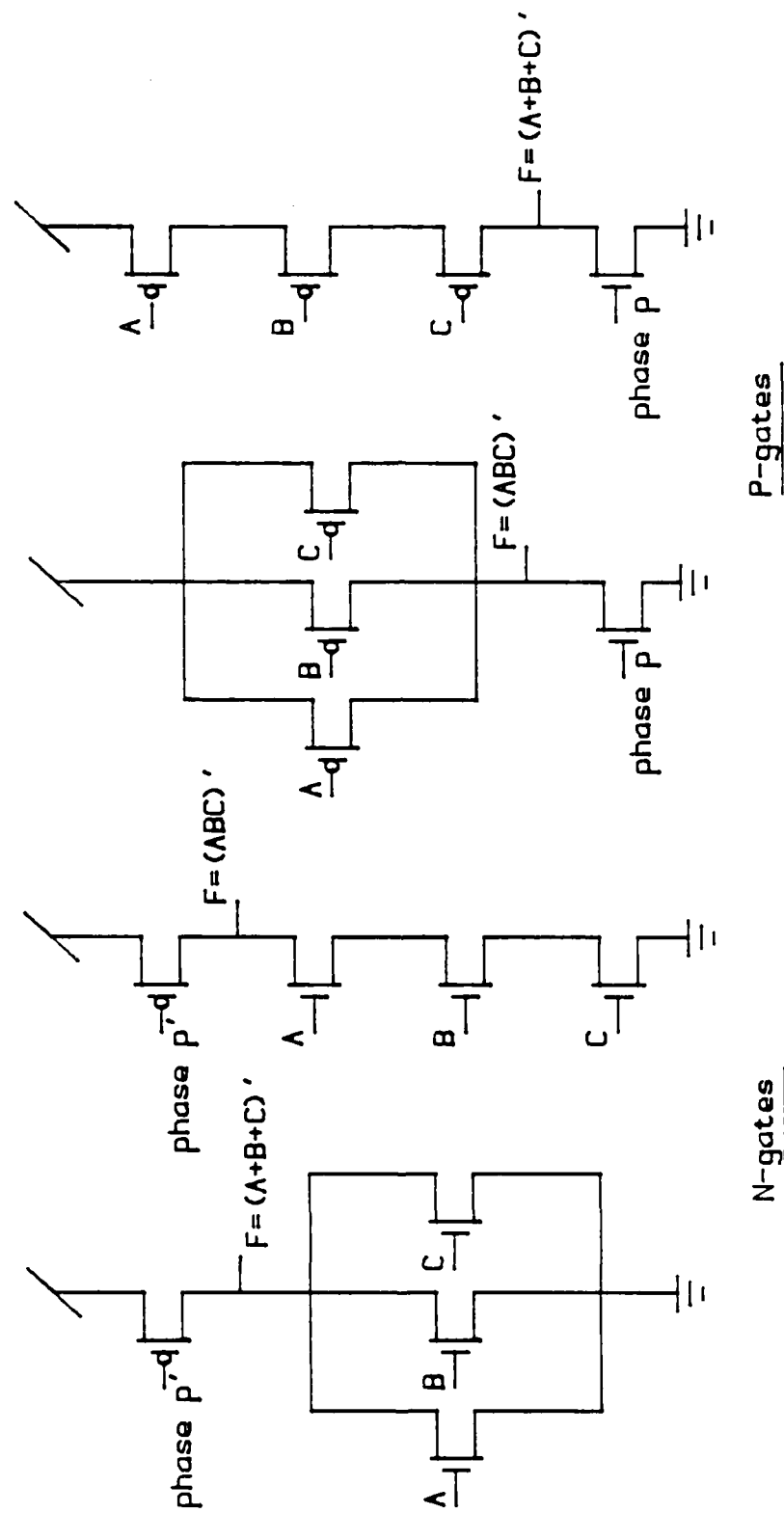


Figure 2.1 Clocked CMOS circuits

series with the logic evaluation block allows precharging without requiring all of the NMOS gates to be turned off. No static power drain is encountered other than from leakage. Applying a high signal to the evaluation device enables this "gated" device.

The dual of a N-gate is a P-gate. The logic is evaluated using PMOS transistors and the precharge (pre-discharge) device is a NMOS transistor. The output is unconditionally precharged low (logic "0") by the NMOS transistor during the precharge period and conditionally charged high by the PMOS transistors during the evaluation period. If the evaluation logic fails to charge the output node, it remains in the precharged state. As in classic CMOS, the series-parallel combination of PMOS transistors must be the dual of the parallel-series combination of NMOS transistors for the same logic function. For example, NMOS transistors in parallel provide the NOR function, whereas PMOS transistors in series form the NOR function. This is illustrated in Figure 2.1.

N-gate can drive or be driven by a P-gate. This characteristic comes from the fact that the evaluation devices of the P-gate should be OFF during the precharge phase, which means that active high gate inputs are needed. This is possible if these gates are driven with N-gate outputs which are precharged high. Therefore a string of

clocked gates of alternating types can be precharged during the precharge phase, and during the evaluation phase logic signals propagate in a ripple through (asynchronous) manner. A notable advantage of this concept is the ease of running successive stages. Addition of an evaluation device to the gates increases logic flexibility (for example, N-gate driving an N-gate), although the asynchronous nature of the circuitry is sacrificed.

## 2.2 Types of CMOS gates

Clocked CMOS gates can be divided mainly into two families: ripple variety without the evaluation device; and the gated variety with the evaluation device. Figure 2.2 and Figure 2.3 illustrate two versions of each variety. Dual versions work on similar grounds. It should be noted that similar series-connected gate versions also exist.

### 2.2.1 Ripple circuits

The first circuit of Figure 2.2 is NRi, which indicates N-gate, Ripple and inverting output. The inputs are precharged low, the output is precharged high, and the logic evaluated is NOR. The second ripple variety is NRn, which indicates non-inverting output. The inputs and output are precharged low, and the logic function is OR. Since the output voltage swing is limited by the device's threshold voltage, this version is in general not useful,

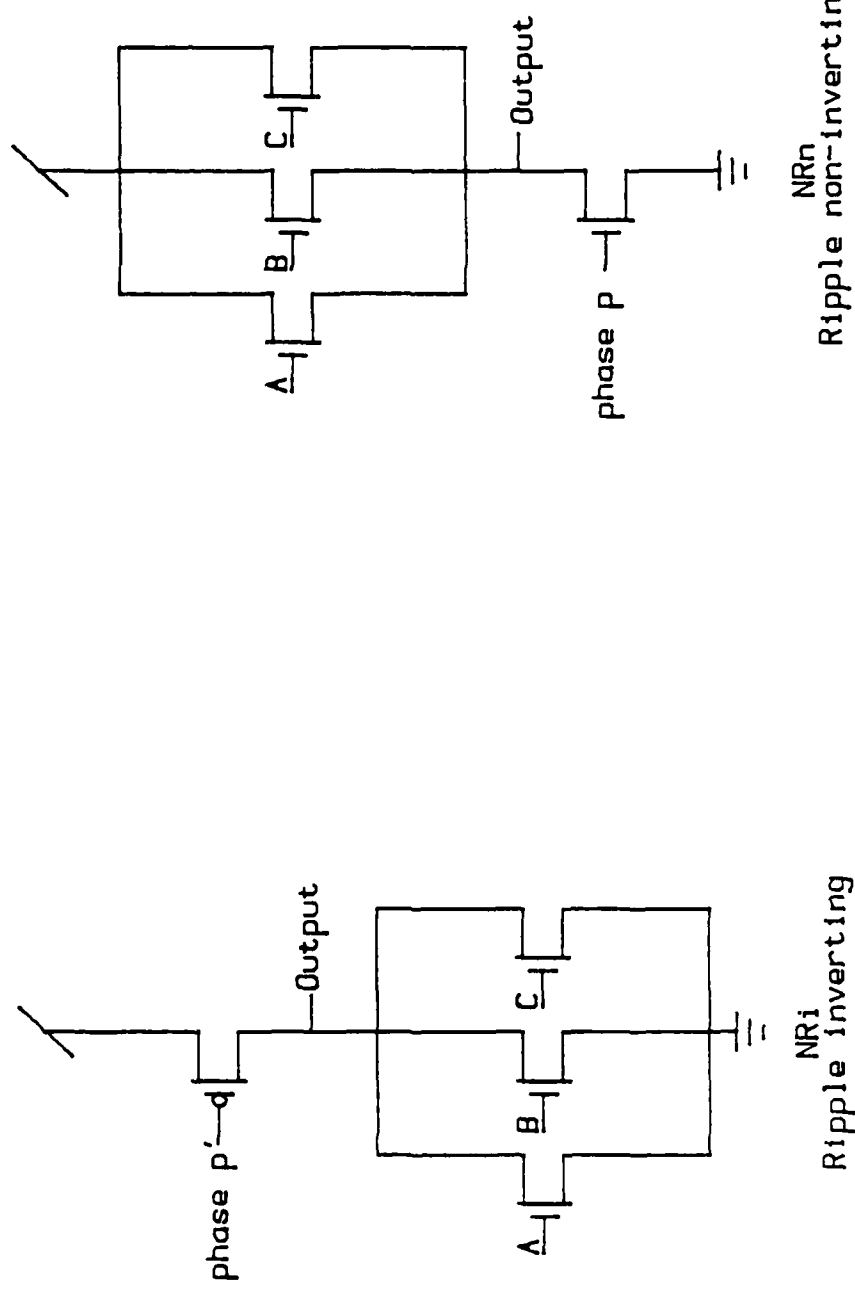


Figure 2 .2 Ripple CMOS circuits

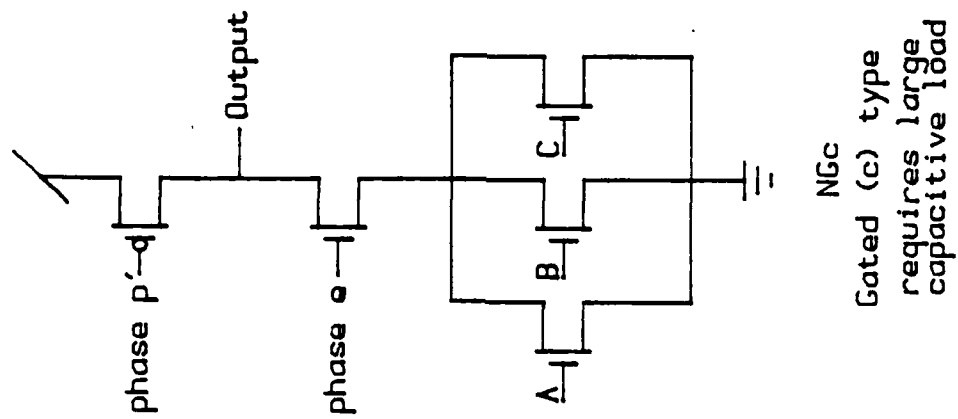
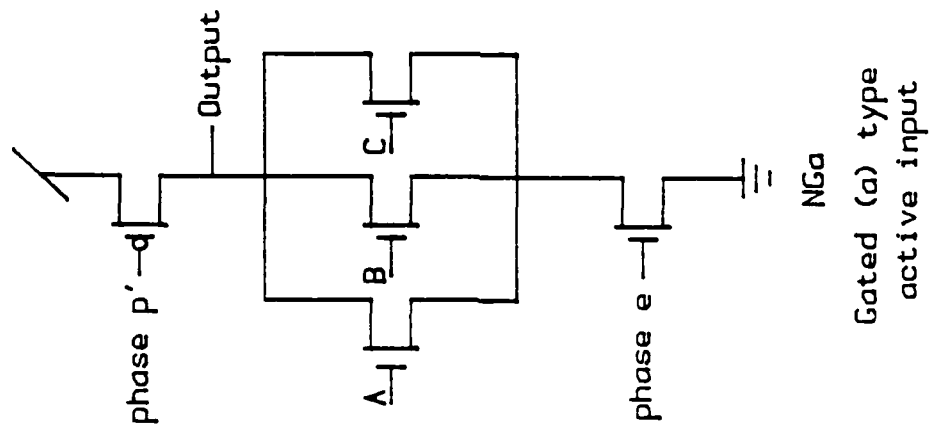


Figure 2.3 Gated CMOS circuits

and is used only in special circumstances.

### 2.2.2 Gated circuits

The difference between the two gated versions as illustrated in Figure 2.3 is the location of the evaluation device. Both of them provide the NOR logic function. NGa version indicates that its inputs must be driven from an "active" sourcing output to prevent charge splitting. That is, the outputs driving this gates must be able to supply current from the positive supply for a "1" input when the evaluation device is turned on. The output of the NGa gate is precharged high during the precharge phase. At this time, even one input precharged high results in the common sources of the logic devices being precharged high to within one threshold below the supply voltage. When the evaluation device is turned on, the source node is pulled to ground with a coupling through the full gate capacitance to the input nodes. If there is no active source for charging this coupling capacitance, then the input signal level is pulled toward ground, thereby reducing the effective drive of the input. A classic inverter or a P-gate can provide the full input drive voltage.

The second gated variety is NGc, which indicates that large capacitance load is required when the fan-in is large, in order to minimize charge splitting. During precharge, the output is precharged high, and the common

drain of the logic devices is presumed to be discharged low. Then if the inputs are discharged low and the evaluation device is turned on, only the output capacitance is available to charge the common drain node. The resulting charge splitting may reduce the output signal to unacceptable levels.

In implementing the PLA, the use of these circuit styles has been explored. While developing the PLA circuits useful for different applications, the concept of N-gates driving P-gates and P-gates driving N-gates was considered.



## CHAPTER III

LARGE PLA3.1 Parallel gates PLA

Large PLA's require large fan-in (F/I). For large fan-ins the speed of many series gates may be disastrous. The alternative is to use parallel gates.

The four basic parallel gate combinations possible for implementing the OR plane in ripple through fashion are:

- 1) NOR      2) OR      3) NOT-NAND      4) NOT-AND

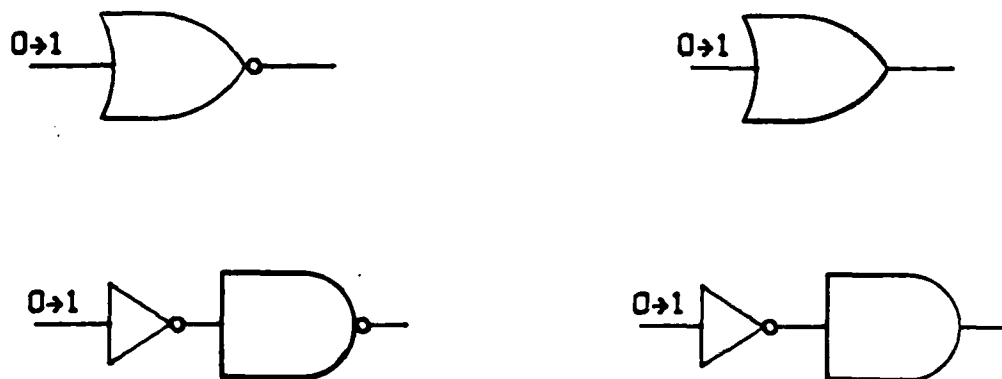


Figure 3 .1 Possible combinations for the OR plane

In Figure 3.1 NMOS devices are the logic evaluating devices for NOR/OR logic and similarly PMOS devices are the logic evaluating devices for NAND/AND logic for parallel configuration. Input gates for NMOS devices should be precharged low and that of PMOS devices precharged high

during precharge phase. From Figure 3.1, it can be seen that all the PMOS transistor gates are driven by inverters; this forces the inverter inputs to be precharged low. Therefore, for a ripple version, all the inputs to the OR plane must be precharged low and conditionally charged high during evaluation.

Figure 3.2 illustrates the four possible ways of implementing AND plane either as a gated or as a ripple variety. They are:

- 1) NAND-NOT    2) AND    3) NOT-NOR    4) NOT-OR-NOT

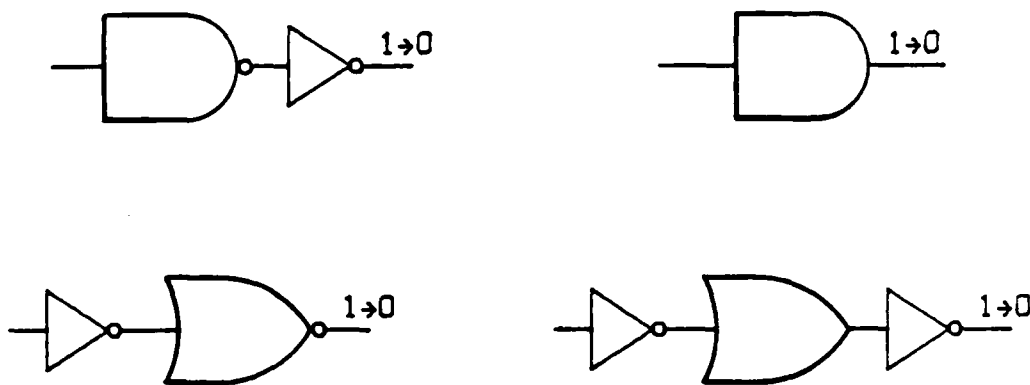


Figure 3.2 Possible combinations for the AND plane

An argument similar to that for the OR plane shows that all outputs must be precharged high and conditionally discharged low during evaluation in contradiction to the OR plane input requirements. Therefore an AND-OR ripple version with parallel devices is not possible. The alternative is to gate the OR plane.

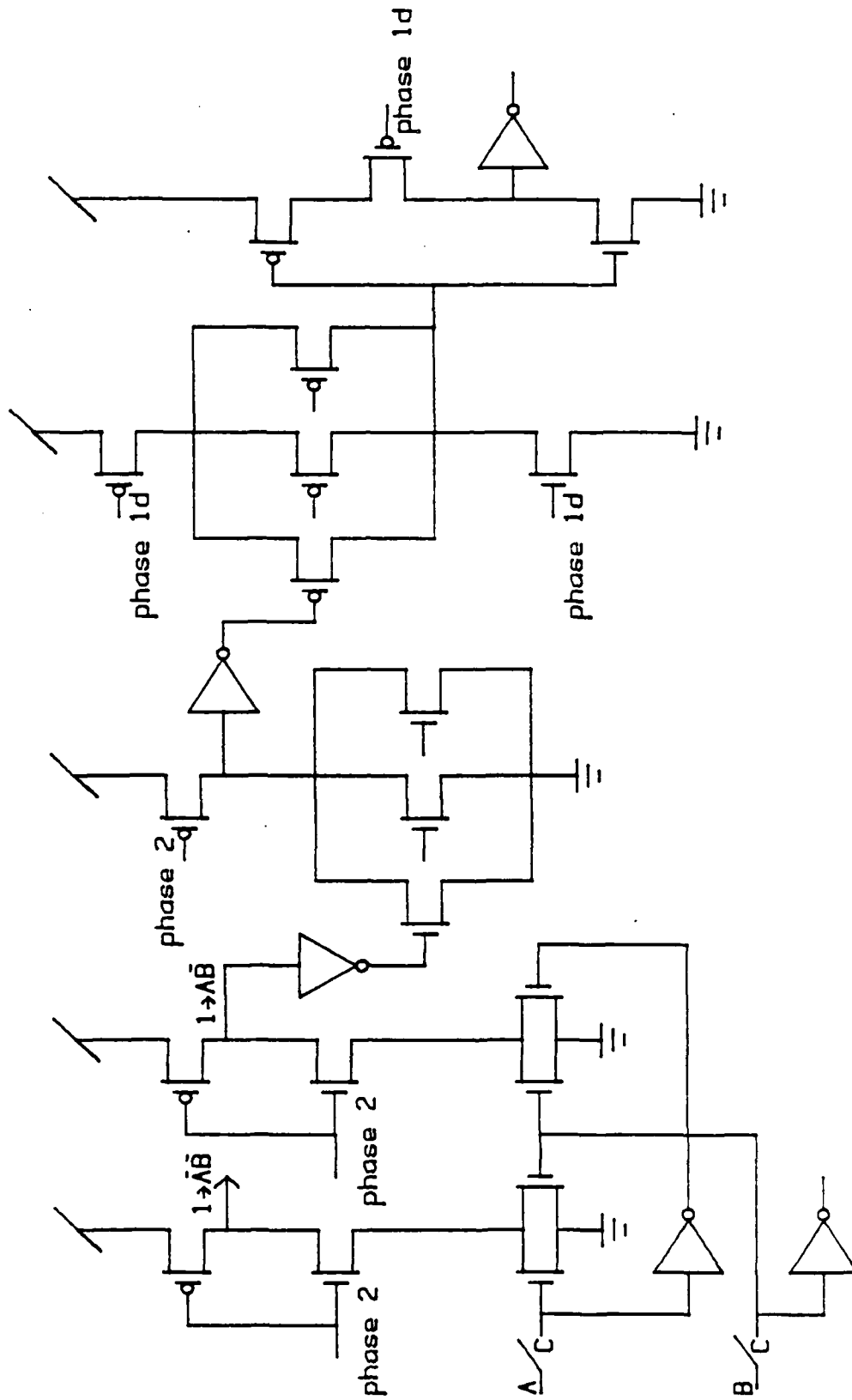


Figure 3.3 Dynamic CMOS PLA  
with input partitioning

One way of implementing the PLA using parallel gates with OR plane gated is shown in Figure 3.3. The timing for this PLA is illustrated in Figure 3.4.

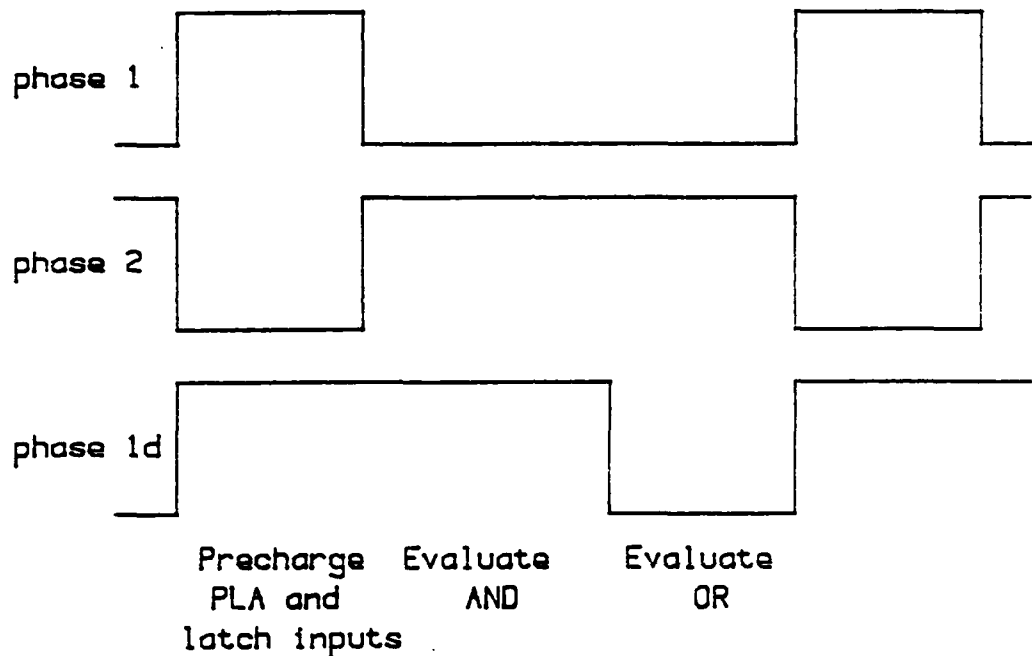


Figure 3.4 Timing diagram of gated PLA

The AND plane is considered to be of N-gate variety and OR plane of P-gate variety. A single master clock is considered for this dynamic PLA. During phase1 the inputs are latched and all the input nodes and output nodes of the AND array and OR array are precharged to their respective standby states. During phase2 (phase1'), the AND plane is evaluated in a ripple through fashion. The clock signal associated with the OR plane is phase1d. In accordance with the PGa requirements, the evaluation transistor should be ON only after the AND plane logic is available at the OR

plane inputs. As the OR plane inputs are precharged low, the evaluation of OR plane before AND plane evaluation leads to confusing results. Phase1d provides the required delay for the OR plane. Then the OR plane logic is evaluated and the information is available in the output flip-flops.

### 3.2 TYPICAL CELLS

#### 3.2.1 Input Circuit

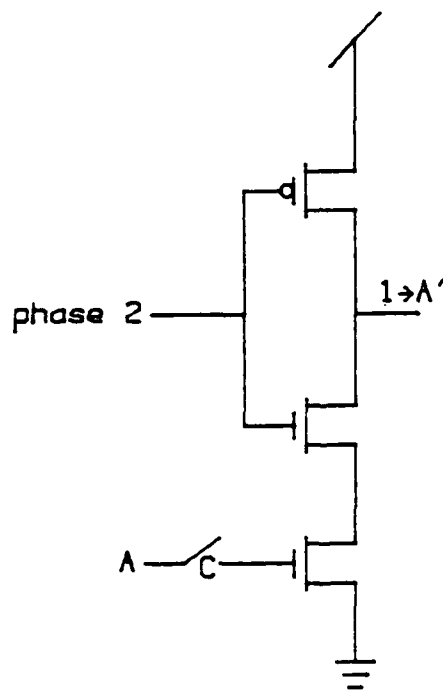


Figure 3.5 Schematic of input latch

The input circuit is shown in Figure 3.5. The switch notation with "C" refers to the CMOS transmission gate which utilizes common-source operation for both the devices to provide rail to rail coupling. The operation of

the switch can be better understood from the two equivalent circuits shown in Figure 3.6. The circuit of Figure 3.6a applies when the incoming signal is high and that of Figure 3.6b applies when the incoming signal is low. Therefore for a CMOS transfer gate, it is always considered that the drain of one device is connected to the source of the other; this enables full voltage swing, a condition not possible with a single polarity transfer gate.

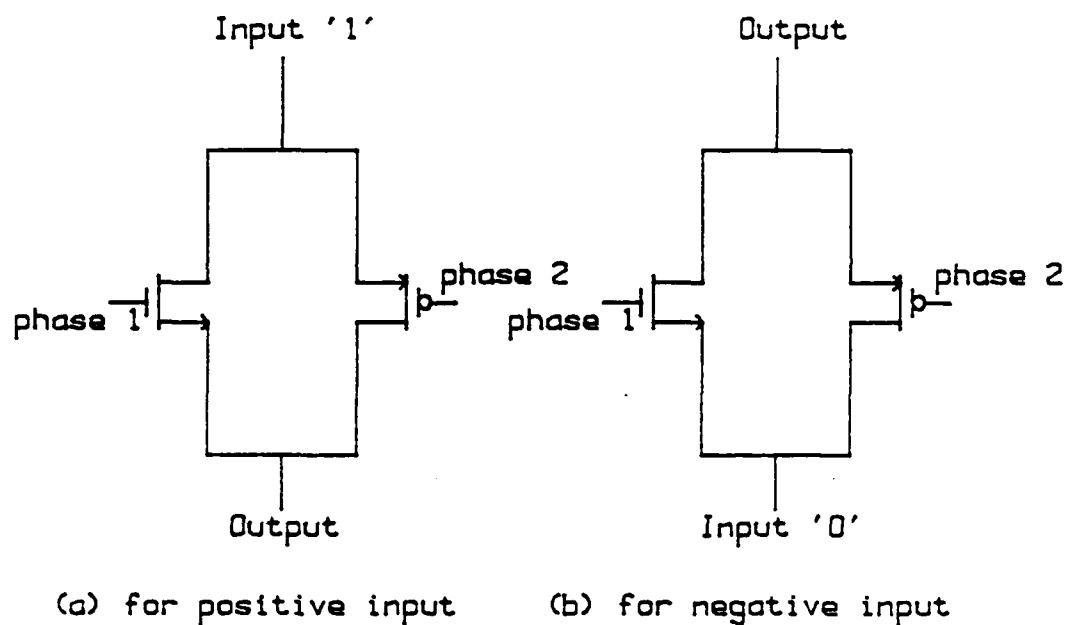


Figure 3.6 CMOS Transmission gate

The rest of the circuit is NGC gate and works on the grounds discussed already. On the falling edge of phase2, the output is precharged high, and it is conditionally discharged low on the rising edge of phase2. The logic available is  $A'$ .

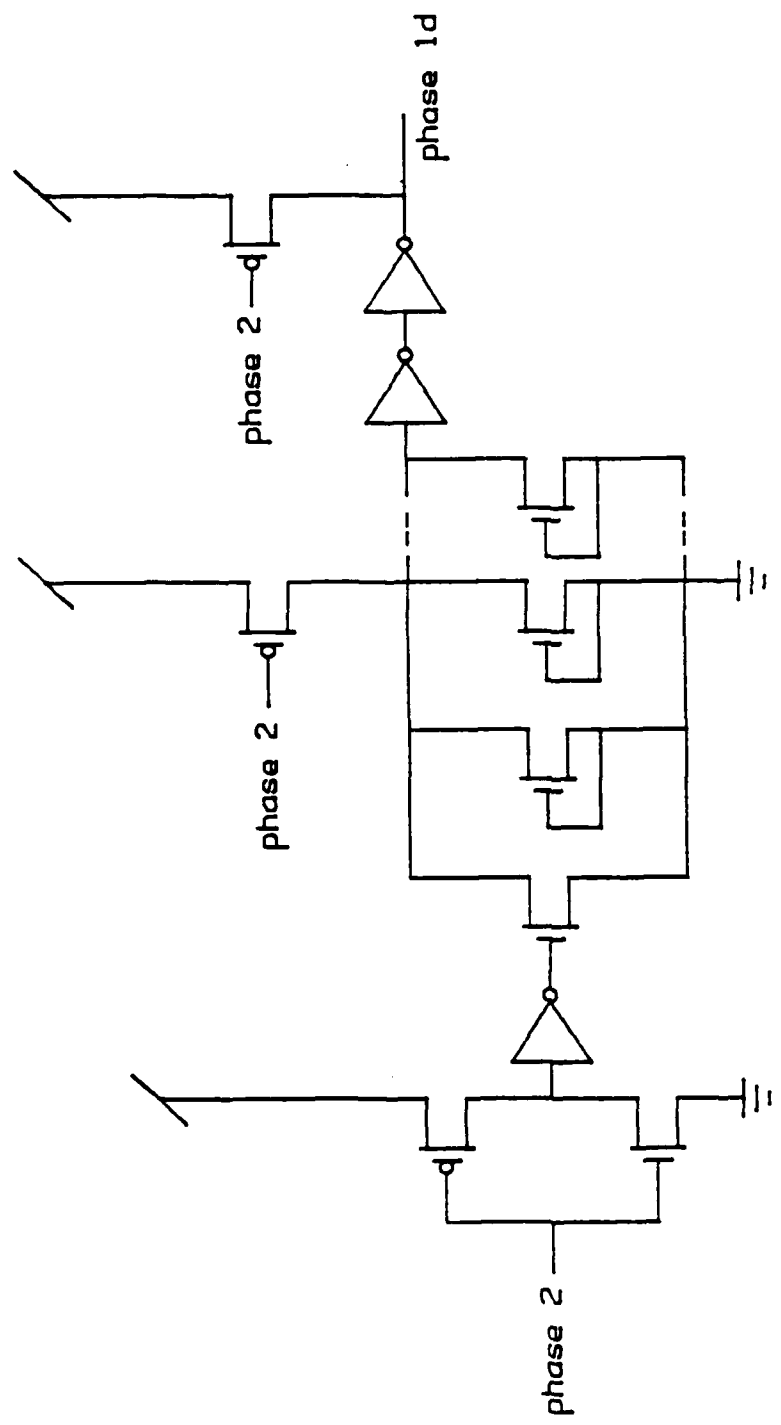
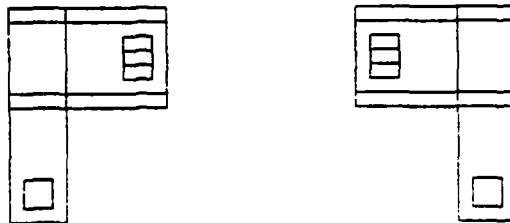
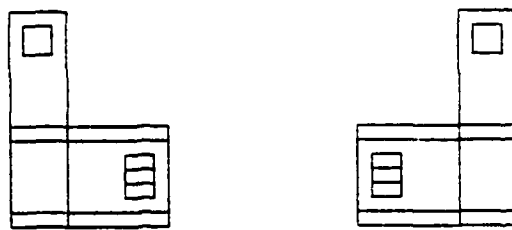
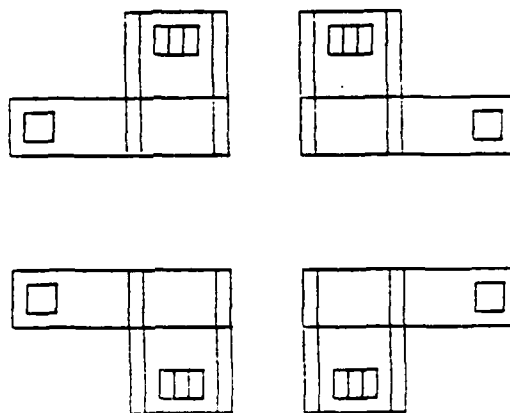


Figure 3 . 16 Delay signal circuit



A-cells



O-cells

Figure 3.15 Diffusion cells programmed into the AND-OR planes.



of the programmed cells, which are appended to the standard PLA layout. Figure 3.14 shows the layout with these transistors programmed in. The diffusion cells which are programmed in, called as Acells and Ocells for the AND plane and OR plane respectively, are shown in Figure 3.15.

### 3.2.3 Delay Circuit

The delay signal phase1d is generated by this circuit. The main functions of the delay signal can be better understood from the timing diagram in Figure 3.4. Signal phase1d is used by the OR plane and output circuit. It should precharge both these circuits when the rest of the PLA is in the precharge phase. So phase1d will be high following the clock signal phase1. When phase1 goes low, it implies the evaluation phase. But phase1d remains high for a little more time until the AND plane is evaluated. Then it goes low for OR plane evaluation and output latching. Again phase1 going high or phase2 going low should pull phase1d high for precharge, thus repeating the cycle.

The circuit used to generate a signal satisfying the above requirements is shown in Figure 3.16. All the NOR inputs except one are connected to ground so that all these transistors are OFF at all times. This way the output is delayed without affecting the logic. Thus a worst-case

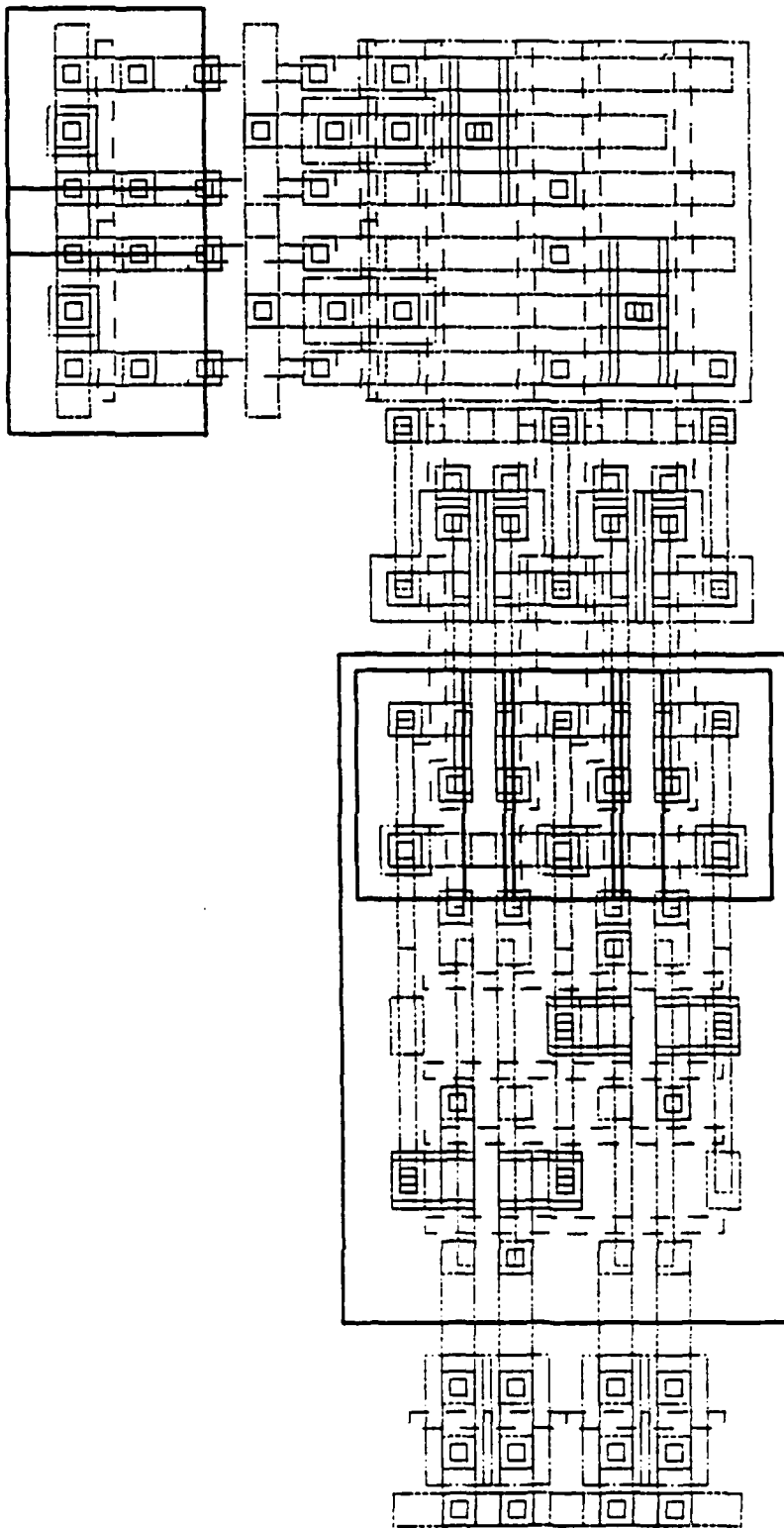


Figure 3.14 AND-OR layout with active areas programmed.

### 3.2.2.1 Programming the PLA

At every location in the AND plane, there are three possibilities: a transistor which is turned on for a logic "1" input, a transistor which is turned on for a logic "0" input, and no transistor at all. These are represented as 1, 0, and x respectively. For every location in the OR plane there are two possibilities: either the presence or the absence of a transistor. They are represented as 1 and 0 respectively. In Figure 3.13 there are no transistors in the AND and OR planes. It should be noted that in the AND plane of Figure 3.13, there are only two inputs and their complements. For example, the AND and OR planes are programmed with the following logic:

AND inputs		OR inputs			
0	0	1	0	0	0
0	1	1	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

The PASCAL program of Appendix A allows programming in diffusion, thereby creating a transistor wherever necessary. The PASCAL program takes the above information as input data and generates MSLL (Mississippi State Layout Language) code which in turn gives the layout description

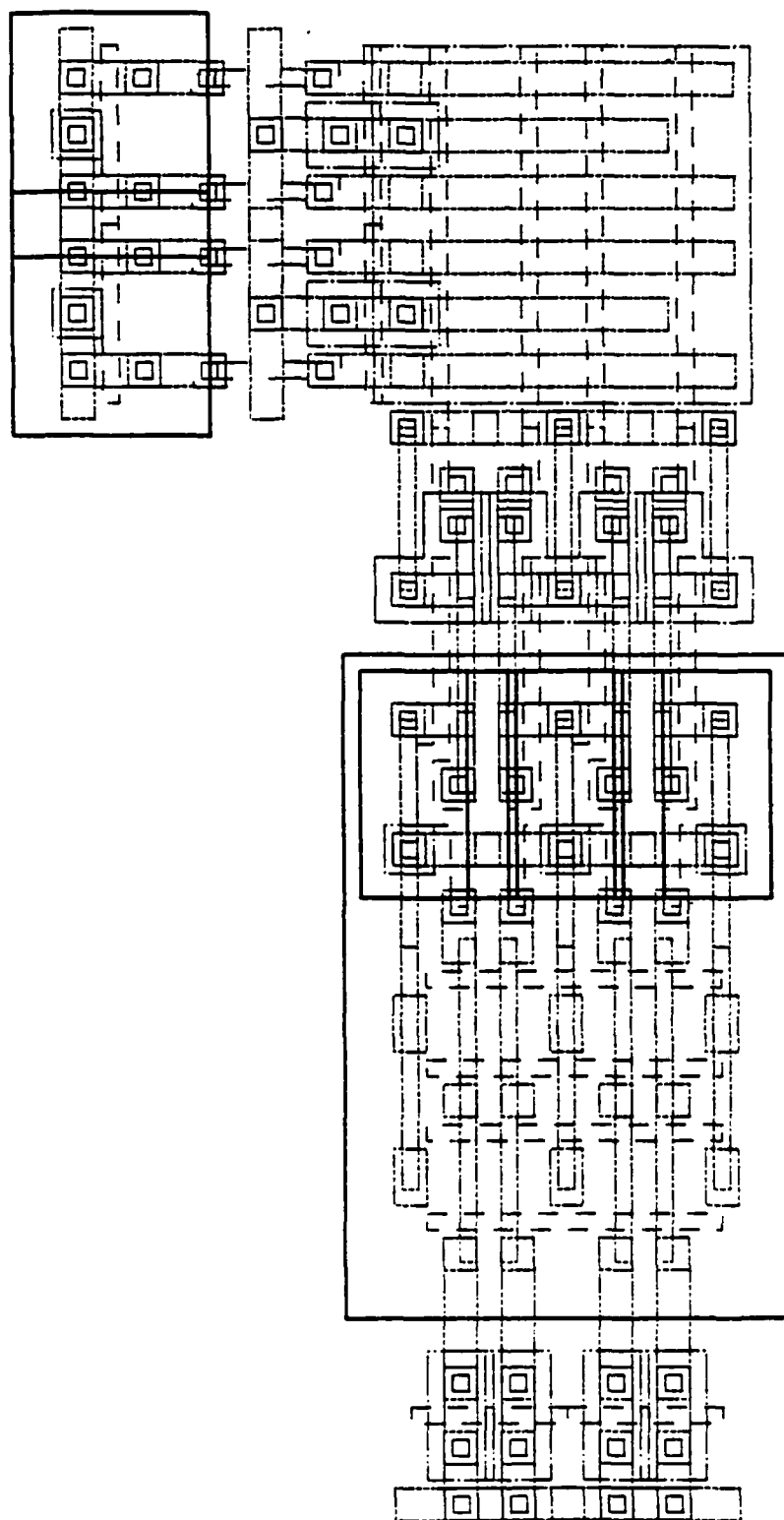


Figure 3.13 Layout of AND-OR Configuration for Large PLA.

both precharged low. Since the inputs are precharged low, and the logic is evaluated by PMOS transistors, the evaluation transistor which is also a PMOS transistor should not turn on until the AND plane logic is available at the OR plane inputs. This signal  $\text{phase1d}$ , which controls the OR plane is generated by the delay circuit. The OR plane outputs feed the output latches. Figure 3.12 shows the schematic of the AND-OR plane. The layout with four inputs, four implicants, and four outputs is shown in Figure 3.13.

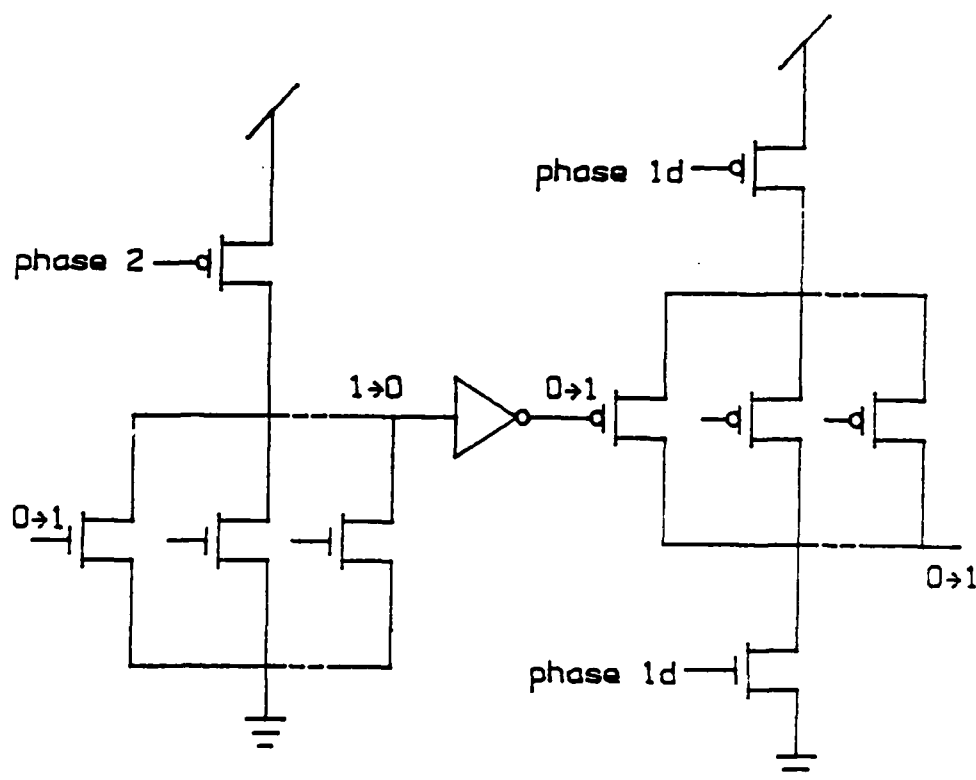


Figure 3 .12 AND-OR circuit

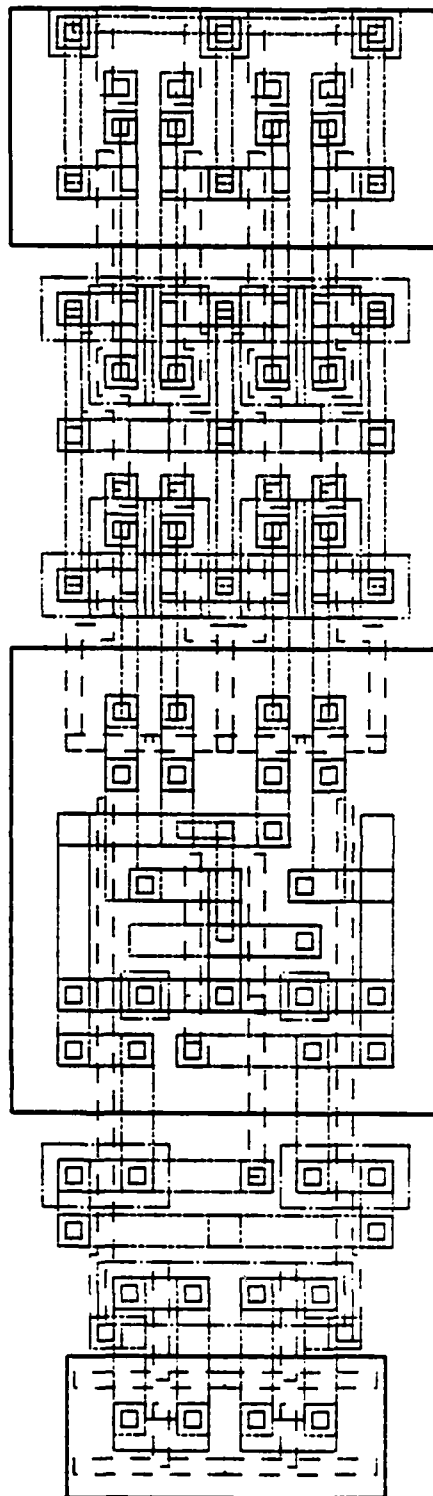


Figure 3.11 Layout of input circuit without input partitioning.

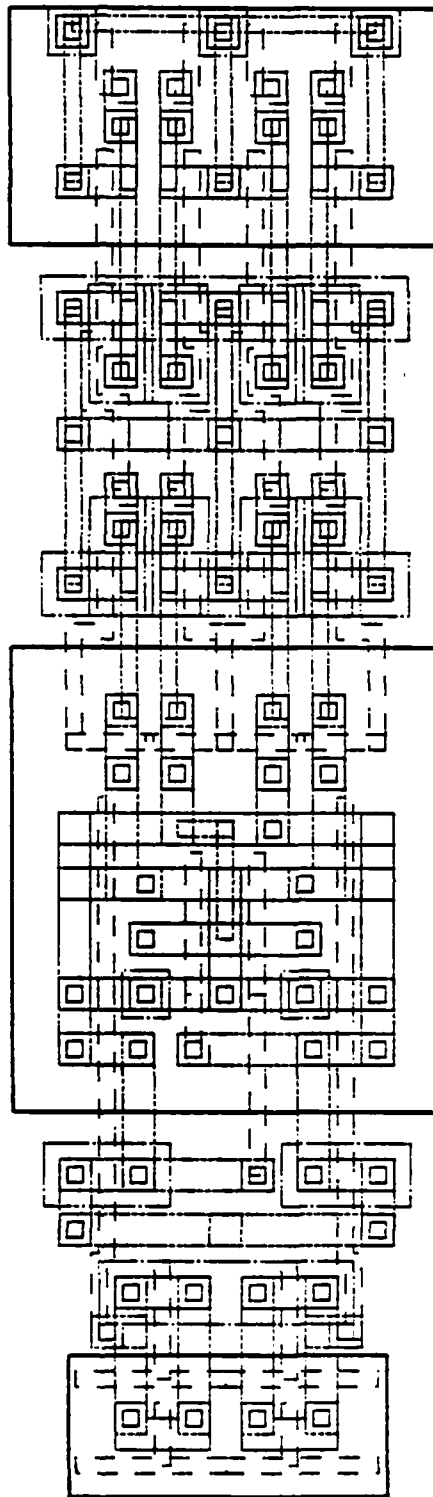


Figure 3.10 Layout of Input circuit with input partitioning.

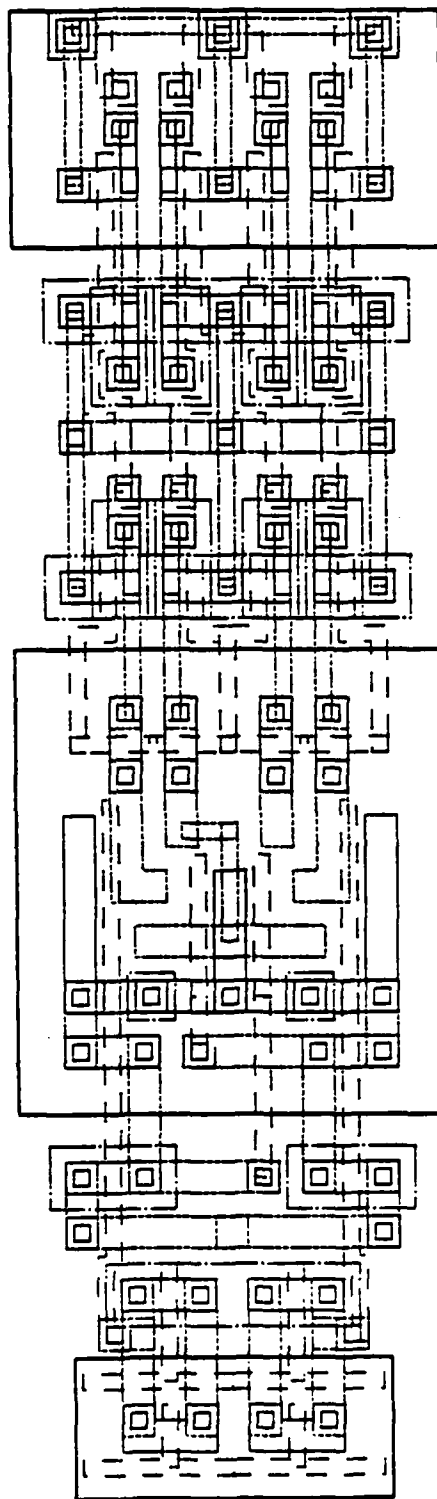


Figure 3.9 Layout of Input Circuit.



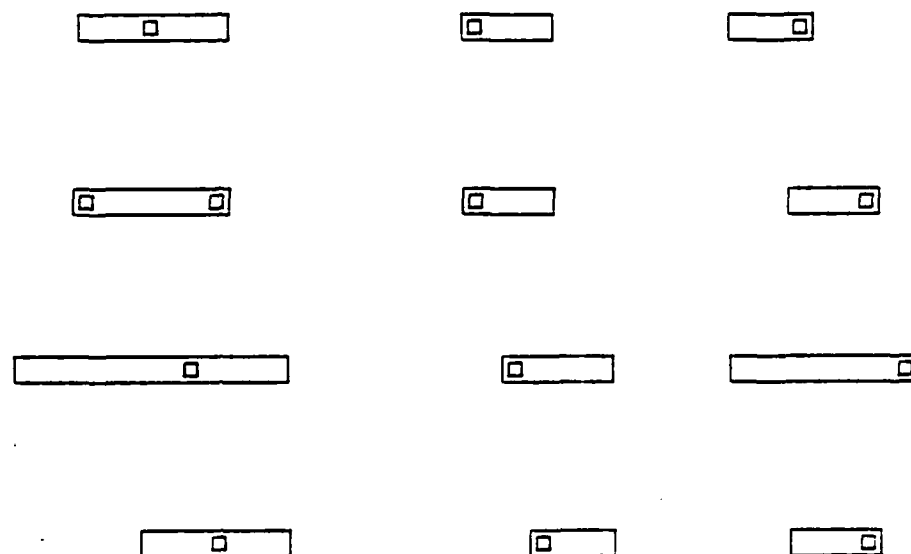


Figure 3.8 Input partitioning ( NOR ) cells.

Active areas are programmed with the program given in Appendix A, so that the circuit provides the combinations mentioned above. Out of the twelve cells termed NOR cells, four are used for each input circuit cell with two inputs. The NOR cells named N1 through N12 are shown in Figure 3.8. The layout of the circuit before the active regions are programmed is shown in Figure 3.9. Figure 3.10 shows the layout with input-partitioning, and Figure 3.11 without input-partitioning.

### 3.2.2 AND-OR

The heart of the PLA configuration is the AND-OR. Of the many possible combinations, the one used for this design is NOR-NOT-NAND. It should be noted that only parallel transistor configurations are used. A NOR in the first phase implies that it should be driven with inverted inputs, thus providing AND logic from NOT-NOR. The NOR is implemented as  $NR_i$ , where the inputs are precharged low and the outputs are precharged high during the precharge phase. Logic is evaluated in ripple-through manner during the evaluation phase, conditionally discharging the output node.

A similar combination of NOT-NAND provides the OR logic. A P-type gated version (PGa) is used to implement the OR. The signal used for the precharge and evaluation devices is  $\phi_{seid}$ . The OR plane inputs and outputs are

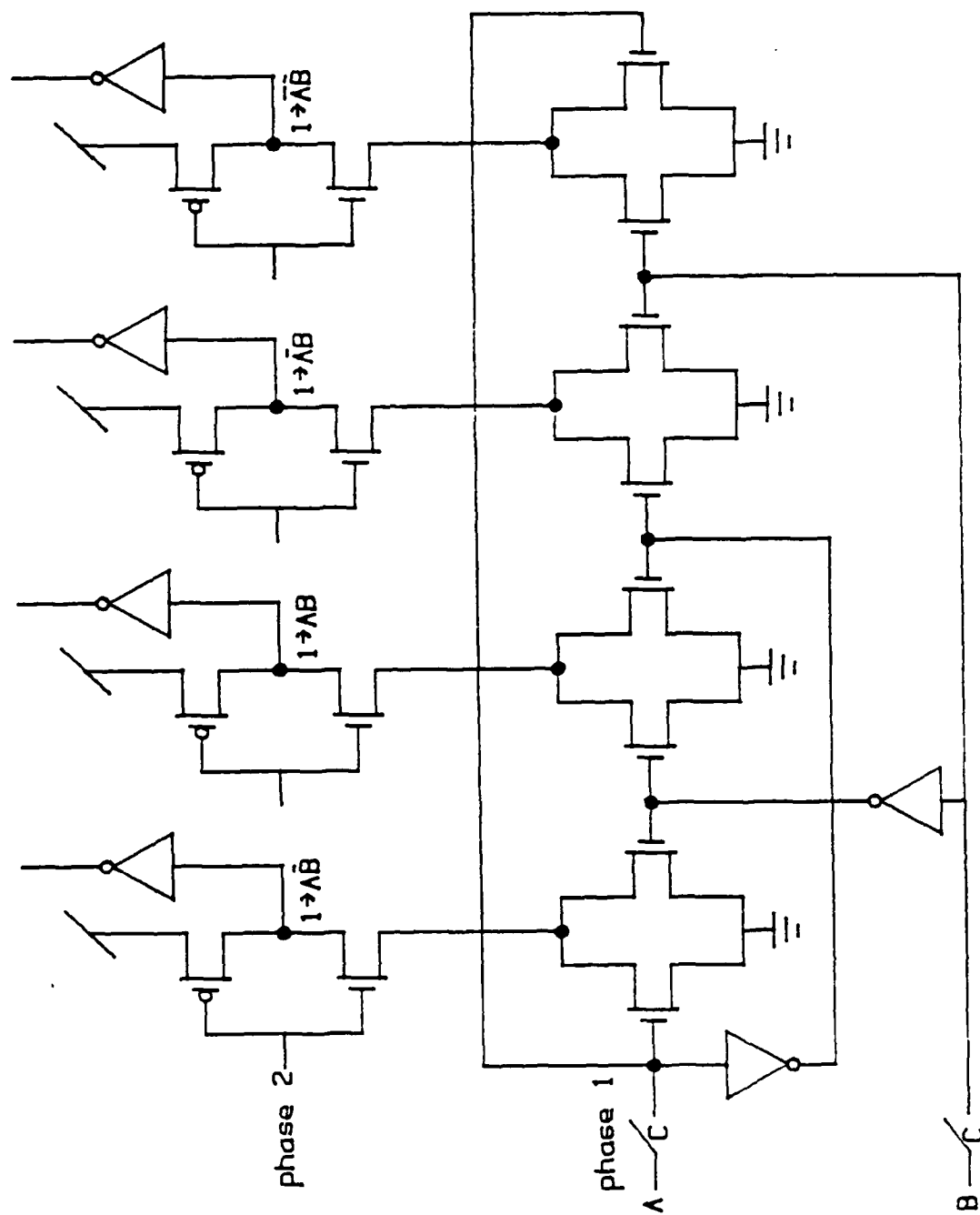


Figure 3.7 Input latch with partitioning

Two input signals can be ANDed before entering the AND plane. This concept termed input-partitioning is used as a means of compacting the design. The input-partitioning can be done with NOR gates in either an NGa or an NGc version; the latter is used for this design. When input-partitioning is done for inputs A and B, then AB, A'B', A'B, and AB' drive the AND plane. It does not cost the AND plane any extra inputs because every input also has its complement driving the AND plane; for two inputs, there are four input lines in the AND plane. The programming capability provides the option of having either the partitioned inputs or the original inputs at the output. The PASCAL program of Appendix A provides the option of using input-partitioning for the design. The circuit with input-partitioning used for this design is shown in Figure 3.7. This Figure shows that for every two inputs there are four outputs, and that every output has three options. For example, at the partitioned output AB, there is a possibility of obtaining AB or A or B by programming. Therefore, there are 81 combinations possible altogether. For simplicity only two combinations are considered for this design.

	<u>Output1</u>	<u>Output2</u>	<u>Output3</u>	<u>Output4</u>
i)	AB'	AB	A'B'	A'B
ii)	A	B	B'	A'

delay path is created. It should be noted that, of all the AND plane outputs this is the most delayed output. This is the dummy row of the AND plane; it is placed at the top of the plane.

After passing through two inverters, phase2 is connected to the only input of NOR which controls the output node. When phase2 is low, this transistor is OFF and its output is precharged high with the PMOS transistor. When phase2 goes high, the output node is discharged to ground. This node after passing through two inverting buffers is given as phase1d signal to the OR plane and the output latch. Phase1d should go high when phase1 goes high, in order to force the OR plane and output latches to precharge mode. If this change in phase1d is expected from the AND plane output, there will be a delay. Therefore, a PMOS transistor with phase2 input is used so that phase1d immediately reacts to the change in phase1. The layout of this cell is shown in Figure 3.17.

#### 3.2.4 Pmostly Latch

The dynamic Pmostly latch is used to latch the OR plane outputs. Its circuit schematic in Figure 3.18 shows that the input must be precharged low. So during precharge, the logic "0" input turns the NMOS transistor OFF, but the PMOS transistor will be ON. This PMOS transistor is in series with another PMOS transistor with

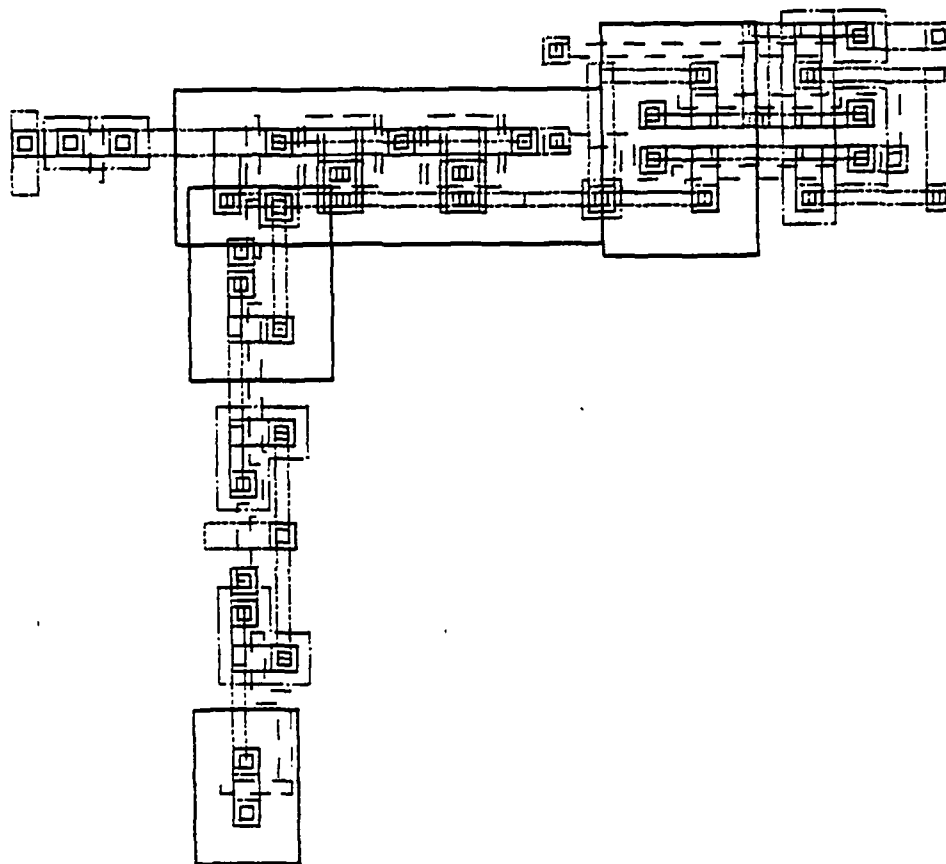


Figure 3.17 Layout of delay signal circuit

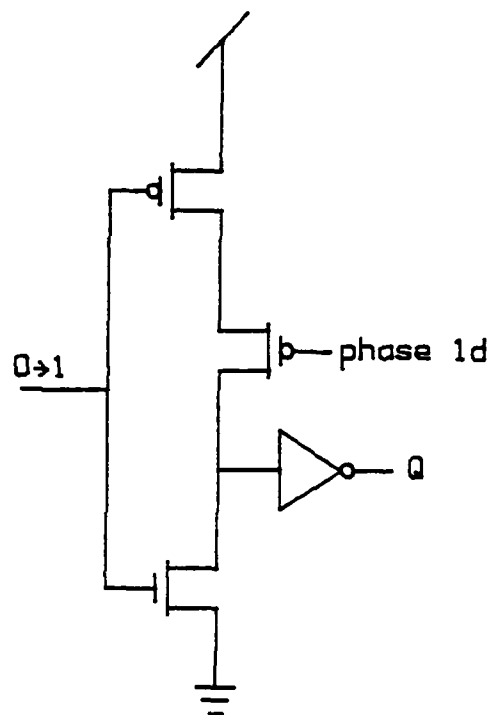


Figure 3.18 Pmostly latch

phase1d input. During precharge, the phase1d signal is high, turning off the PMOS transistor. Hence during precharge, there is no path to the output. During evaluation, if the input is conditionally charged high, the NMOS transistor turns on and a valid output is available. If the input stays low during evaluation, the PMOS transistor is ON and the series PMOS transistor with phase1d input is also ON as phase1d is low during evaluation. Thus a valid output is available in either case during evaluation. The output of the PLA is tapped at the inverting buffer output because the previous node is a

floating node. Since the additional PMOS transistor controls the latch, this latch is named a Pmostly Latch. The layout of the latch for two inputs and two outputs is shown in Figure 3.19.

### 3.2.5 SR-Latch

The schematic of the Set-Reset latch is shown in Figure 3.20. The back to back inverters are driven by NMOS transistors. The aspect ratios of these NMOS transistors should be sufficiently large compared to the internal PMOS transistors so that the internal latch can be upset. For the present design, the aspect ratio of NMOS transistors is twice as large as that of PMOS transistors. The set and reset transistor inputs come from the OR plane, and both of them are turned off during the precharge state. For setting or resetting, the respective input should go high during evaluation, enabling the latch. It should be noted that no timing signal is required here, but two OR plane outputs are required for each flip-flop. Output drive buffers are used to supply adequate source current. Figure 3.21 illustrates the layout for the set-reset latch.

All the cells described above are documented in Appendix C with their circuit schematics and layouts. These cells are used in the design of the PLA example in the following section.



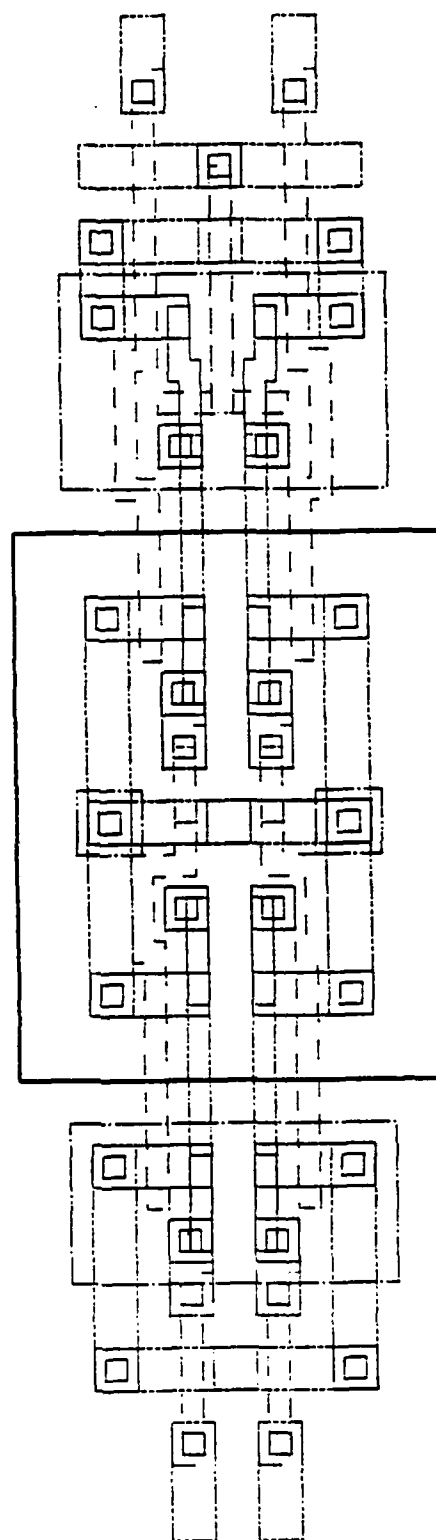


Figure 3.19 Layout of dynamic Pmostly latch.

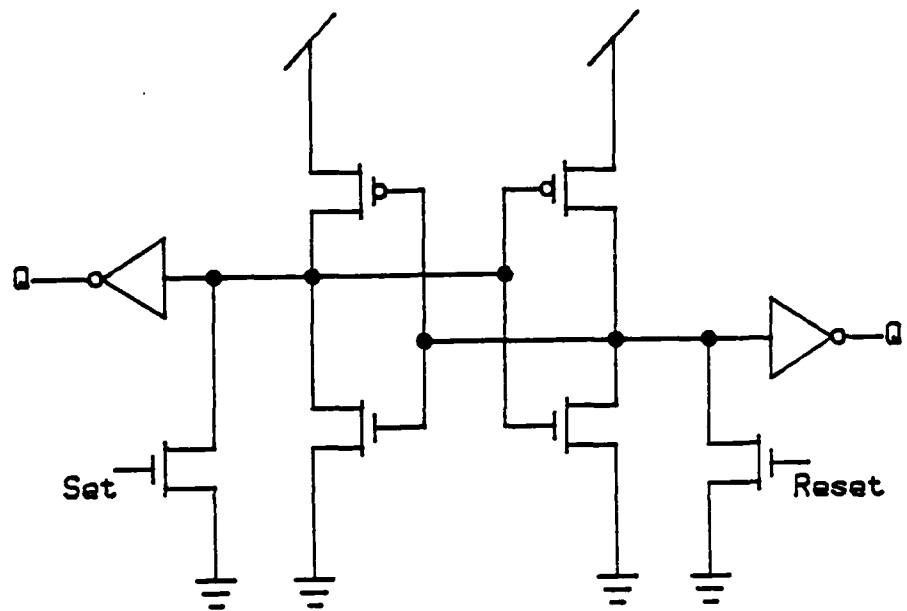


Figure 3.20 Set-Reset latch

### 3.3 The PLA example

This PLA has been designed with 32 inputs and 28 outputs. Either of the output latches can be used depending on the application. The inputs to the PLA will be represented as I1 through I32, and outputs as O1 through O32. The examples programmed into the PLA to test the working of the PLA are explained in the following sections.

#### 3.3.1 Priority Encoder

A priority encoder which has 16 input lines and four output lines is considered. Its truth table is shown in Table 3.1. Inputs I6 through I21 and outputs O9 through O12 of the PLA are used for the priority encoder.

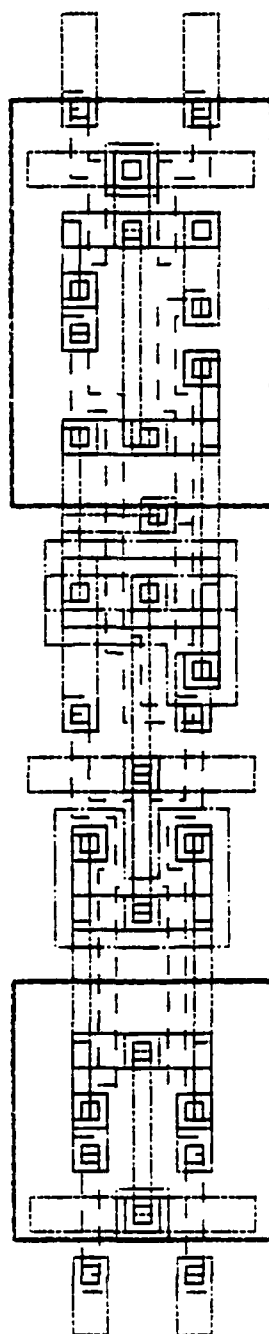


Figure 3.21 Layout of set-reset latch

### 3.3.2 Seven Segment Display

The description and the truth table for a seven-segment display is shown in Table 3.2. The seven-segment display needs four inputs and seven outputs for its operation. Therefore inputs I2 through I5 and outputs O2 through O8 of the PLA are used.

### 3.3.3 Test Signals

It is also important to note the best and worst cases of the PLA. Test signal (a) is generated by applying a logic "1" input at I1 and collecting at O1 using the bottom implicant. The test signal (b) is generated by feeding logic "1" to all the 32 inputs and taking the output from O28 by using the top implicant. Test signals (a) and (b) correspond to the best and worst cases, respectively. With these test signals, the propagation delays in the AND and OR planes are estimated for the best and worst cases.

### 3.3.4 Programming

Input-partitioning can be done on every two inputs, so there are 16 sets which can be partitioned. For the present application, input-partitioning is used only on one set, i.e. on inputs I3 and I4. The input data for the PASCAL program to code this information is given below in Table 3.3.

Inputs																Outputs			
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	A	B	C	D
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1
x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	1	1	0
x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	1	1	1
x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	1	0	0
x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	1	0	1
x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	1	1	0
x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	1	1	1
x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	1	1	1
x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	1	1	1
x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	1	1	1
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	1	1	1
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	1	1	1

Table 3.1 16 by 4 priority encoder

BCD Code				Seven-segment Code						
X1	X2	X3	X4	A	B	C	D	E	F	G
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	0	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
0	0	0	0	1	1	1	1	1	1	0

Table 3.2 Seven segment display

o/p1	o/p2	o/p3	o/p4
A	B	B'	A'
AB'	AB	A'B'	A'B
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'
A	B	B'	A'

Table 3.3 Data for input partitioning

The input and output truth tables for the PLA, which include the examples discussed above, are shown in Table 3.4 and Table 3.5 respectively. Tables 3.3, 3.4, and 3.5 are given as input data to the PASCAL program. The program gives a listing of all cells required in MSL format. This is then appended to the rest of the MSL code to generate the complete layout. Layouts of input latches, output latches, AND array, and OR array are shown in Figures 3.22, 3.23, 3.24, 3.25 respectively.

The floor plan of the PLA, including the I/O buffers, is shown in Figure 3.26. The pin assignments of the 36-pad frame are shown in the figure. Nine pads are multiplexed

for use as input during phase1 and output during phase2. Eight pads are left unconnected. Inputs I1 and I22 through I32 are connected to the power supply and outputs O13 through O27 are not connected since they are not used. The interconnections to the pads are illustrated in Figure 3.26. The final PLA chip layout is shown in Figure 3.27.

[illegible]

**Table 3.4 Table of inputs to the PLA**





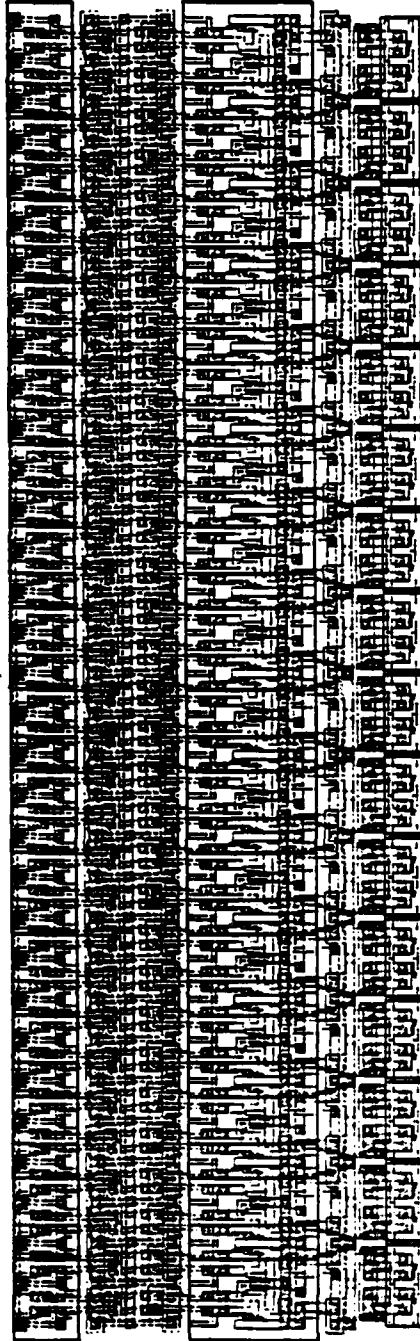


Figure 3.22 Layout of input latches for Large PLA

AD-A158 367

BULK CMOS VLSI TECHNOLOGY STUDIES PART 1 SCALABLE CMOS  
DESIGN RULES PART 2. (U) MISSISSIPPI STATE UNIV  
MISSISSIPPI STATE DEPT OF ELECTRICAL E..

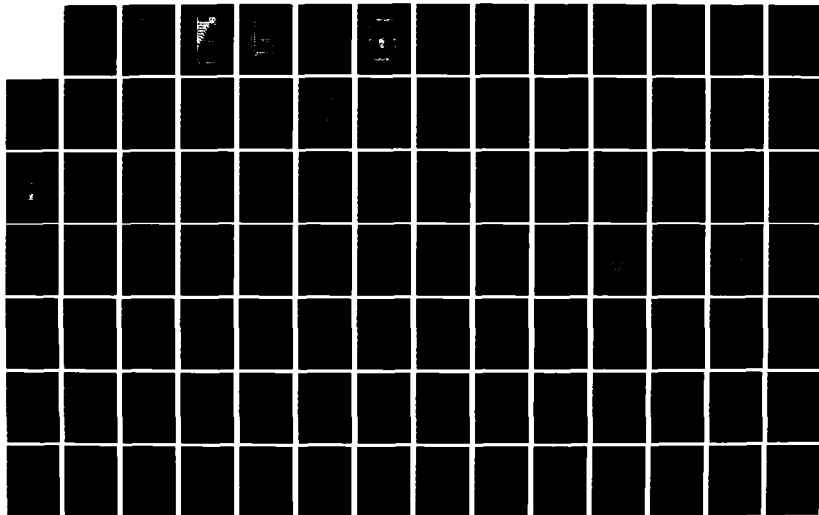
2/3

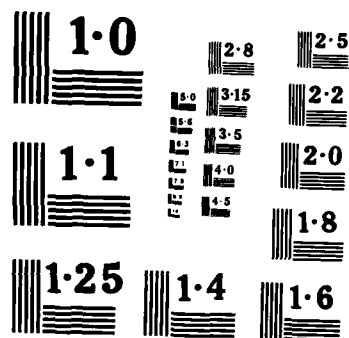
UNCLASSIFIED

J D TROTTER ET AL. 17 JUN 85

F/G 9/5

NL





NATIONAL BUREAU OF STANDARDS  
MICROCOPY RESOLUTION TEST CHART

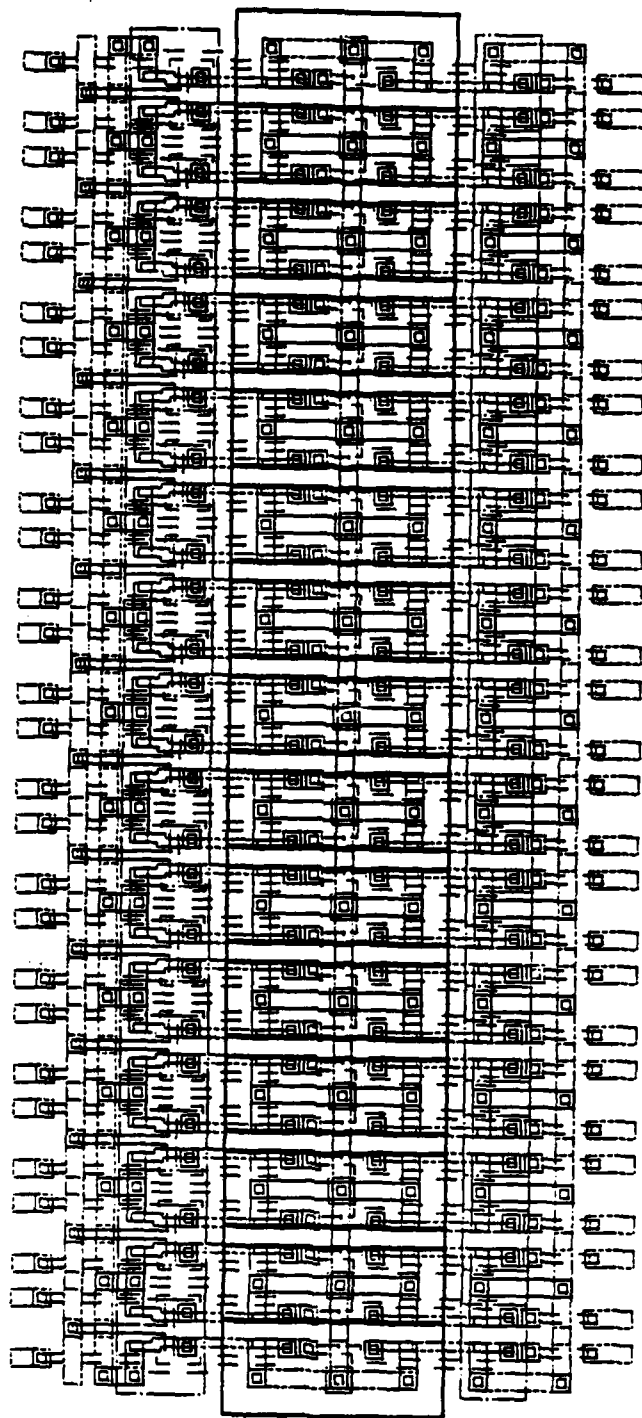


Figure 3.23 Layout of output latches for Large PLA.

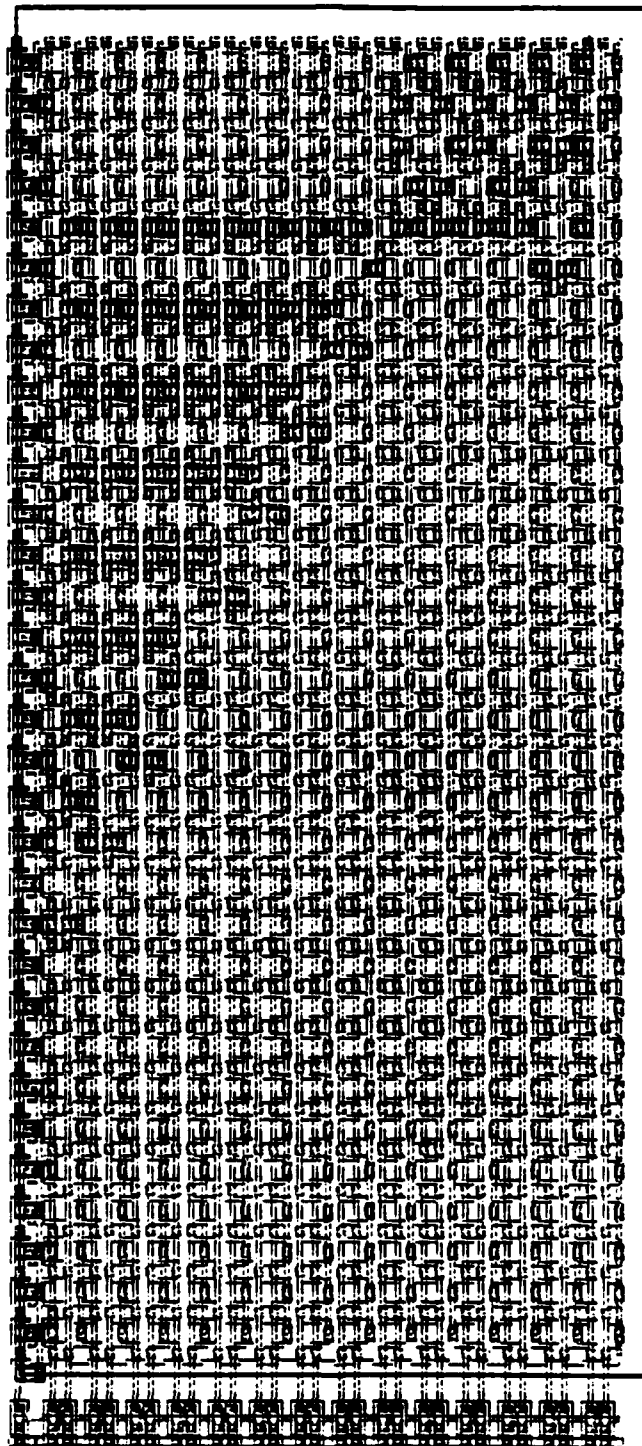


Figure 3.24 Layout of the AND plane for Large PLA.

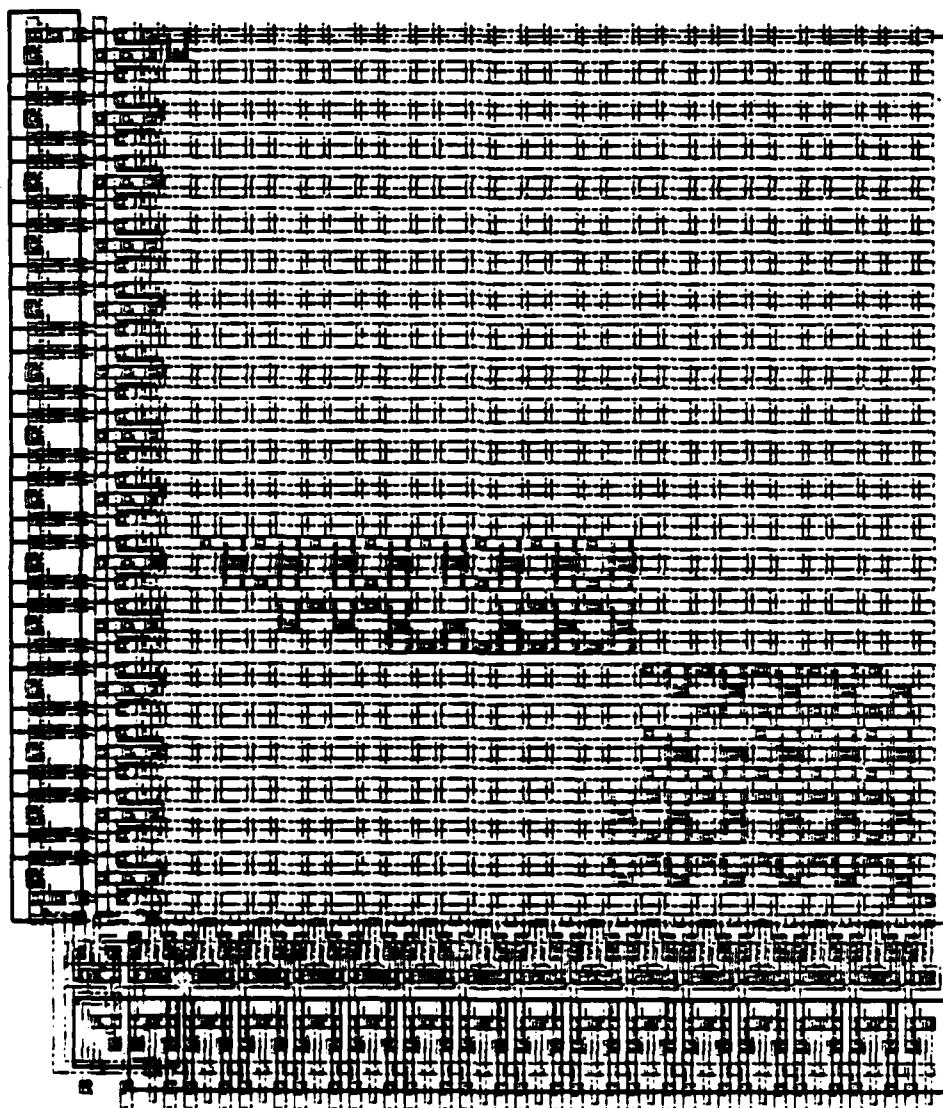


Figure 3.25 Layout of the OR plane for Large PLA.

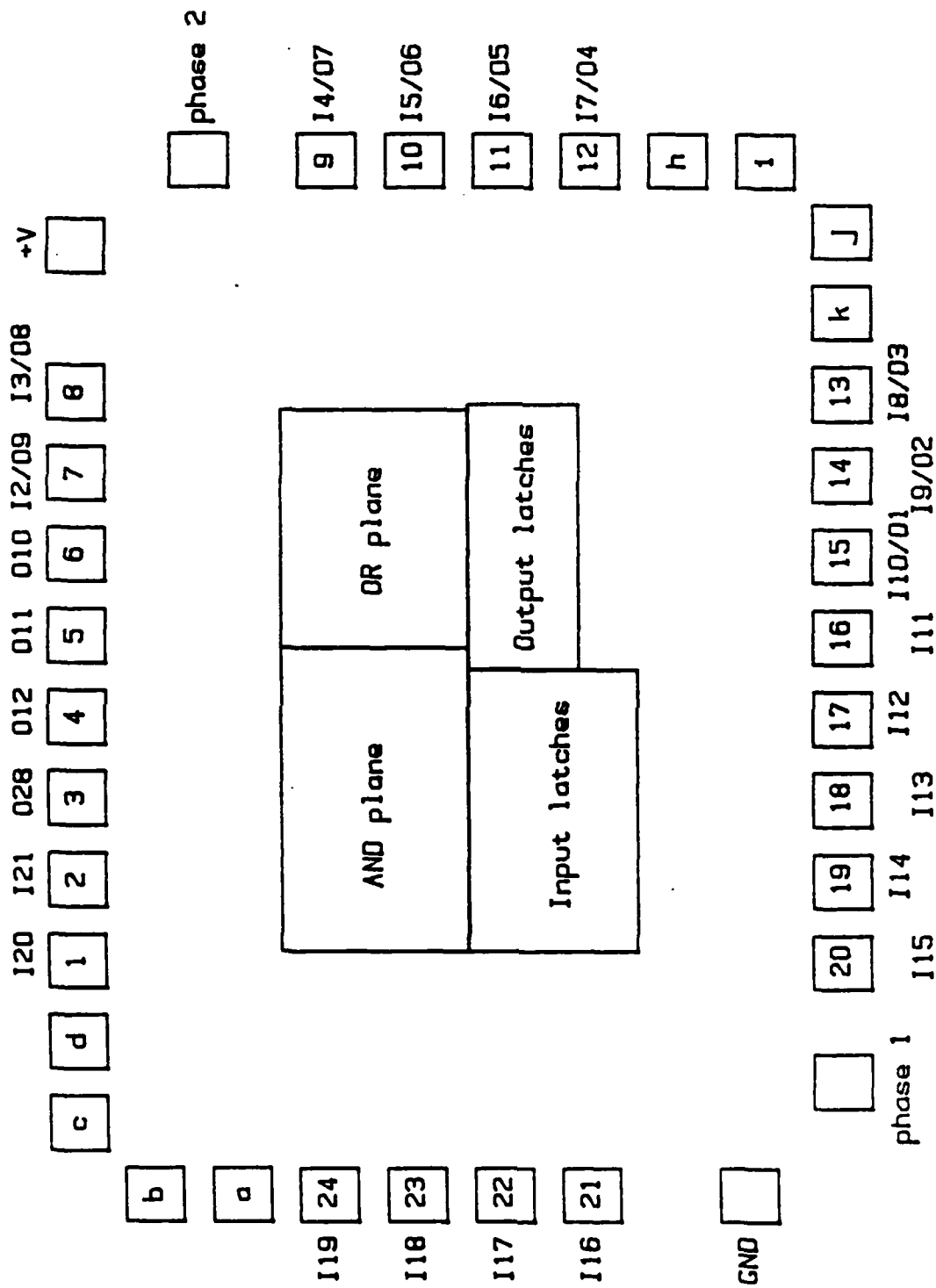


Figure 3. 26 Pin assignments and floor plan of Large PLA chip



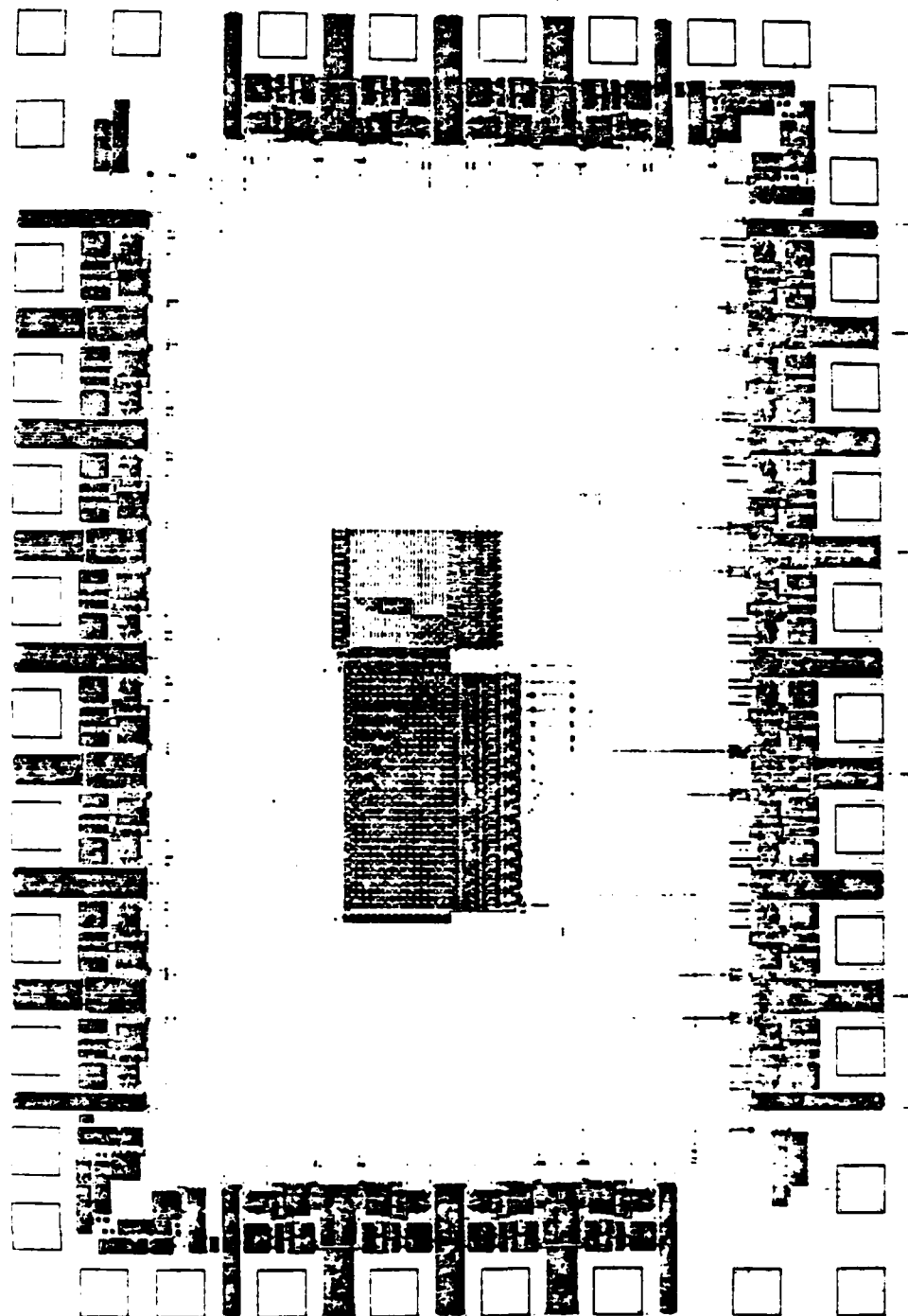


Figure 3.27 The large PLA chip

## CHAPTER IV

RIPPLE PLA'S4.1 Moderate PLA

From the last chapter, it is obvious that it is not possible to have ripple AND-OR circuitry with parallel devices. The alternative is to use series-parallel or parallel-series combinations. Transistors in series give rise to fan-in problems. A way of improving the fan-in and still preserving the ripple nature will now be considered. It will be called a "moderate" fan-in PLA. The logic diagram of the AND-OR is shown in Figure 4.1.

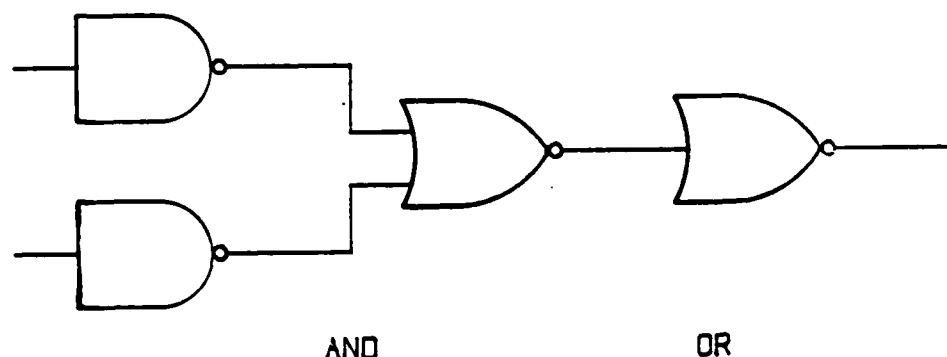


Figure 4 .1 AND-OR of Moderate PLA

The logical AND is implemented in two stages, NAND driving NOR to give AND logic. Suppose there are two inputs, A and B, driving one NAND gate and two inputs, C

and D, driving another NAND gate. The outputs of the NAND gates will be  $(AB)'$  and  $(CD)'$  respectively. The NOR of these two is  $((AB)' + (CD)')' = ((AB)')' \cdot ((CD)')' = AB \cdot CD$ , thus giving the AND function. The circuit schematic for this is shown in Figure 4.2. Series NMOS transistors are used to generate the NAND, and series PMOS transistors are used to generate the first NOR. During phase 1, all the input and output nodes are precharged to their respective standby states. The inputs of the NAND gates should be precharged low and conditionally charged high. If four inputs are used per each NAND gate, four such NAND gates provide the flexibility of using 16 inputs, whereas having 16 transistors in series in one stage is not possible. The layout of the AND plane with 16 inputs is shown in Figure 4.3. The Figure shows the technique used for layout design. The OR plane is implemented with NOR having NMOS transistors in parallel. Its inputs are precharged low, and its outputs are precharged high, as seen in Figure 4.2. The layout of the OR plane with two implicants and two outputs is shown in Figure 4.4.

The input latch shown in Figure 4.5 is used for this PLA. The external inputs are latched during phase 1, since the C switch is closed. In accordance with the requirement of the AND plane inputs, the output is precharged low. The layout of input latch with two inputs is shown in Figure 4.6.



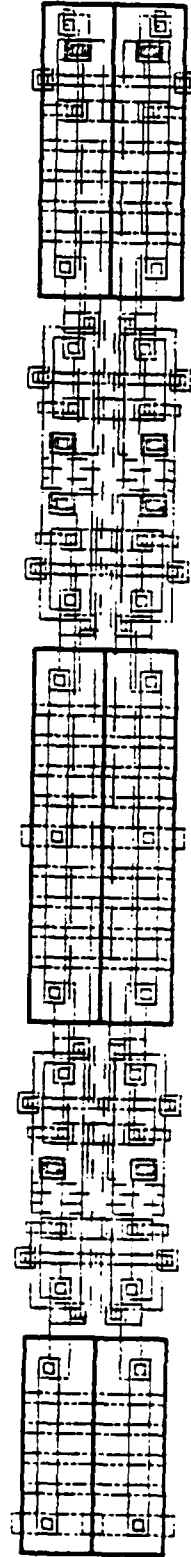


Figure 4.3 Layout of AND plane for 2 implicants

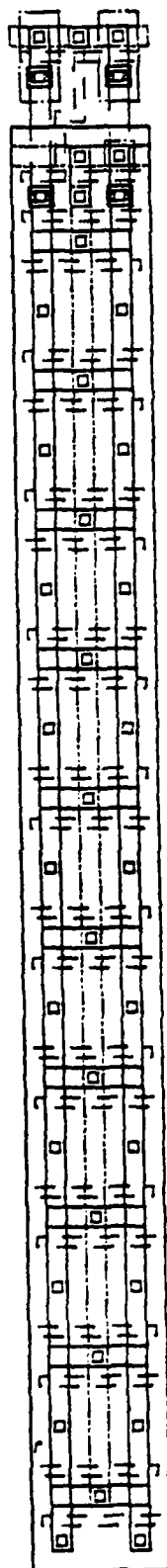


Figure 4.4 Layout of OR configuration for Moderate PLA

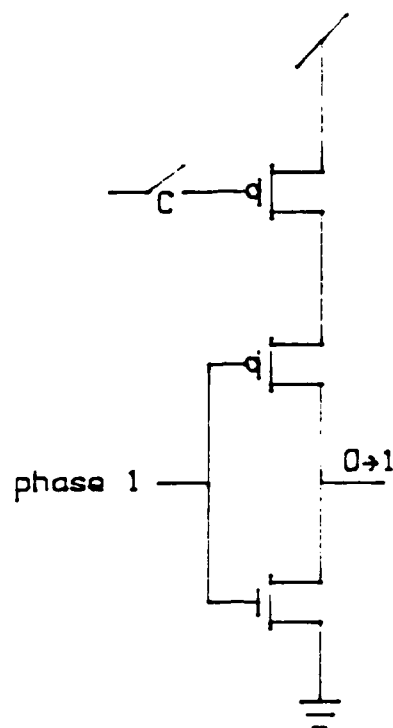


Figure 4.5 Gated input circuitry

The output latch shown in Figure 4.7 is the Nmostly latch. The dual version of it was discussed in the previous chapter. During phase1, which is the precharge phase of the PLA, its input is precharged high. A high input turns the PMOS transistor OFF and turns the immediate NMOS transistor ON, but the series NMOS transistor with phase2 input is OFF. Therefore the output node is floating during the precharge phase. During evaluation, the input is conditionally discharged low and the PMOS transistor turns ON conditionally. If the input remains high, there is still a valid output because phase2 is high during evaluation and thus the series NMOS transistor turns ON.

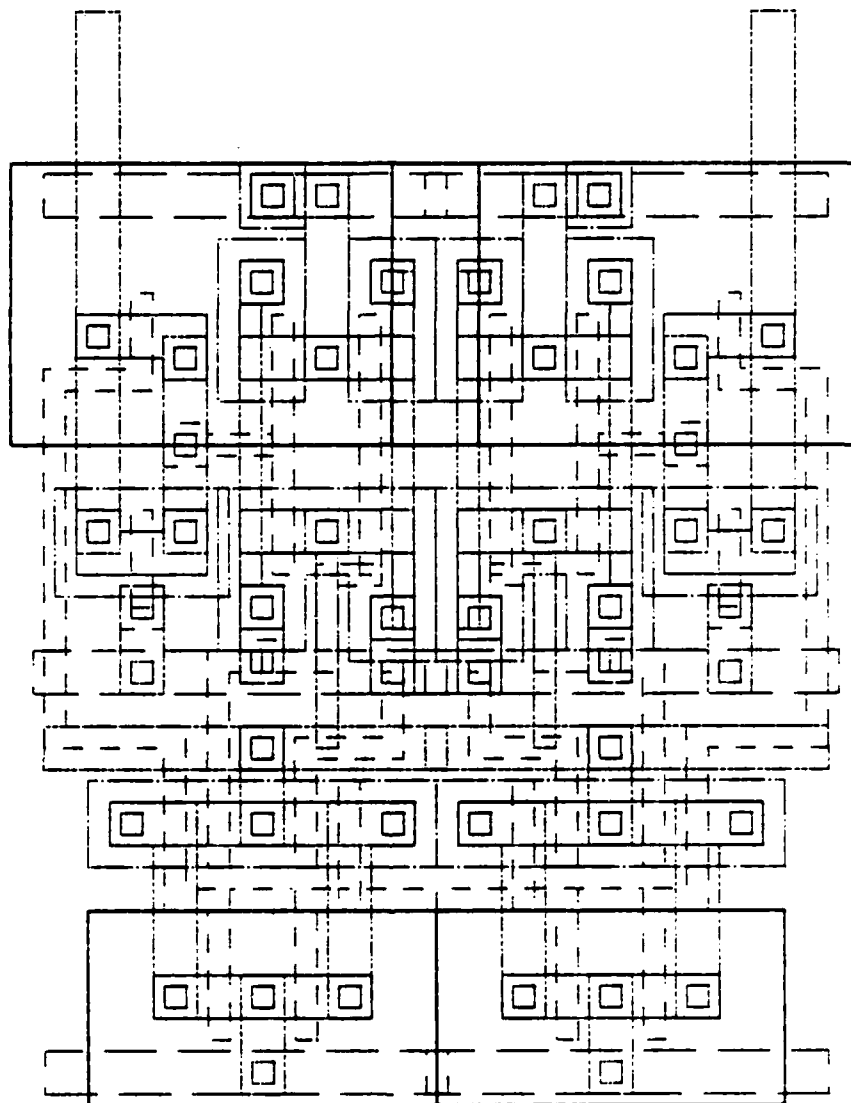


Figure 4.6 Input latch for Moderate PLA



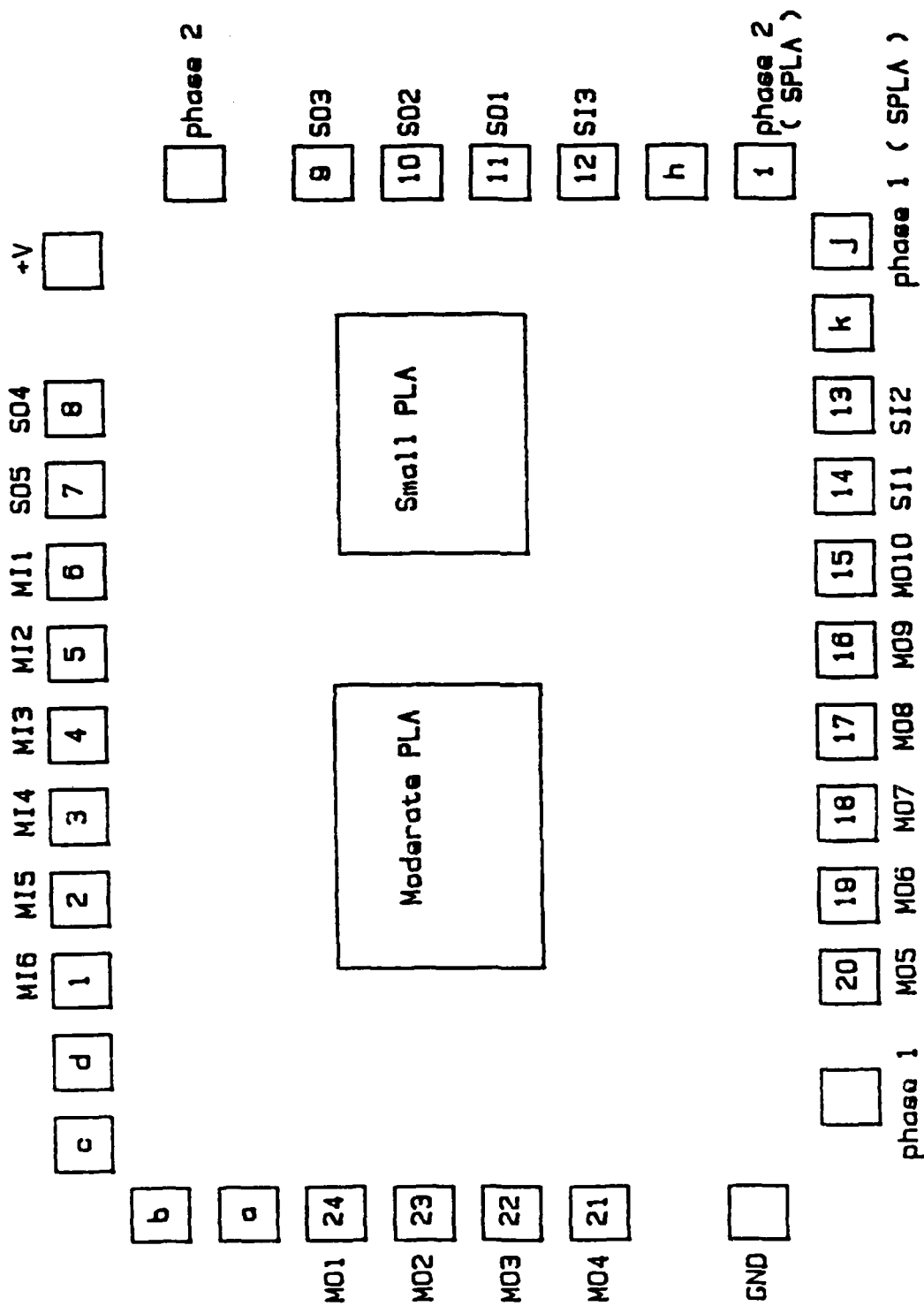


Figure 4. 16 Pin assignments of the chip

<u>Inputs</u>					<u>Outputs</u>						
			Present State		Next State						
C	TL	TS	Yp0	Yp1	Yn0	Yn1	ST	HL0	HL1	FL0	FL1
0	x	x	0	0	0	0	0	0	0	1	0
x	0	x	0	0	0	0	0	0	0	1	0
1	1	x	0	0	0	1	1	0	0	1	0
x	x	0	0	1	0	1	0	0	1	1	0
x	x	1	0	1	1	1	1	0	1	1	0
1	0	x	1	1	1	1	0	1	0	0	0
0	x	x	1	1	1	0	1	1	0	0	0
x	1	x	1	1	1	0	1	1	0	0	0
x	x	0	1	0	1	0	0	1	0	0	1
x	x	1	1	0	0	0	1	1	0	0	1

Table 4.3 Traffic Light Controller

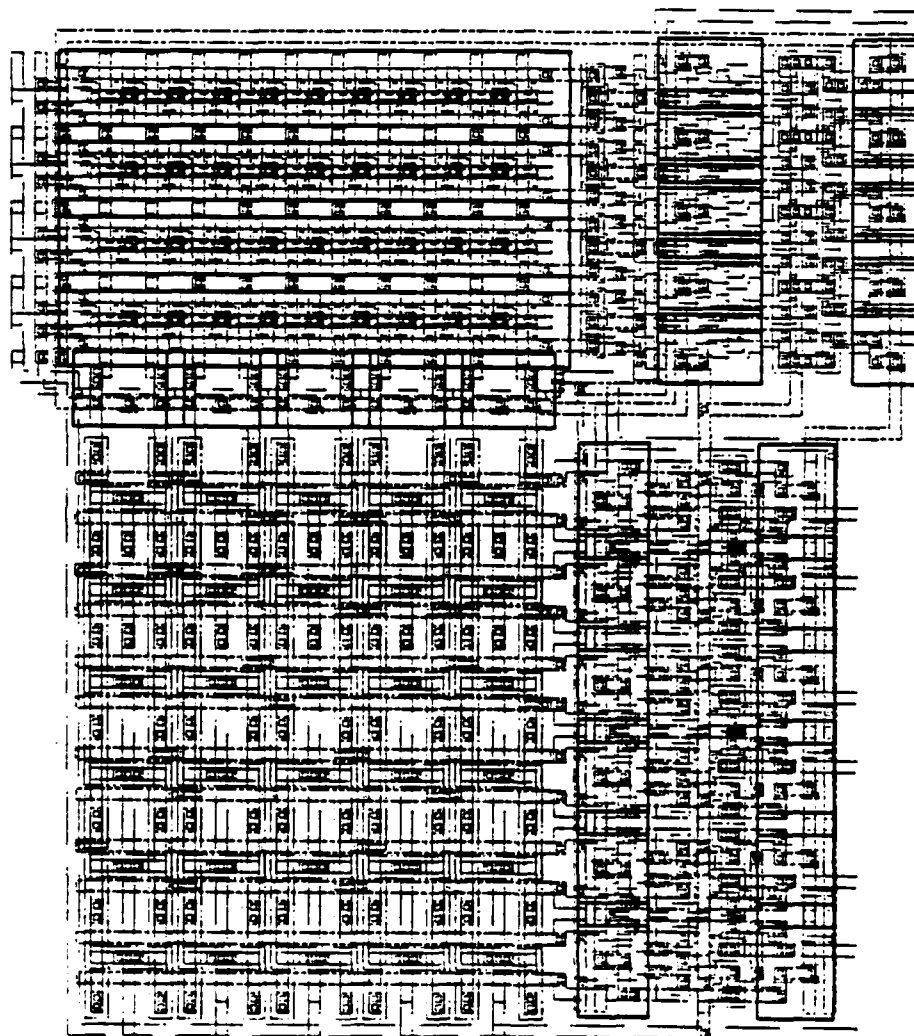


Figure 4.15 Layout of the Small PLA

03 through 07 serve as the controller outputs. The cells of this PLA are included in the cell library of Appendix C. The small PLA, with the example mentioned above programmed in, is shown in Figure 4.15.

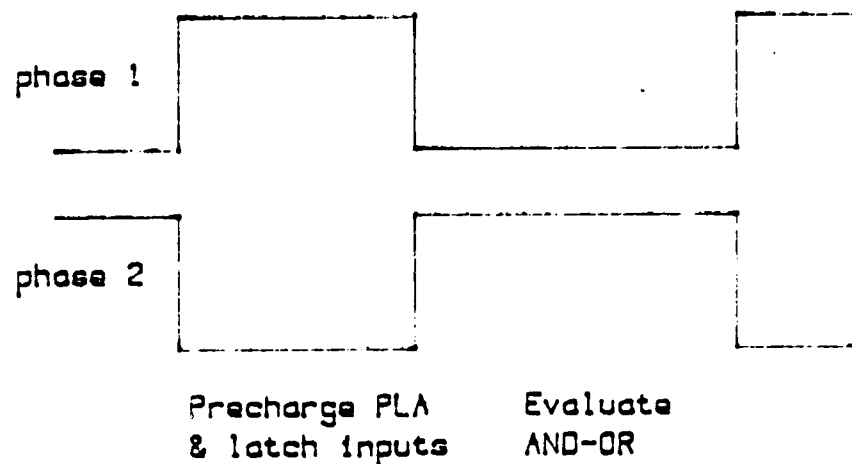


Figure 4.14 Timing diagram of small PLA

#### 4.3 The chip frame

A 36-pad frame similar to that of the previous chapter is used here. The moderate PLA and the small PLA are both fitted into this frame. The small PLA uses eight pins and the moderate PLA uses 16 pins. The two clocks (phase1 and phase2) provided by the frame are used for the moderate PLA. The clocks for the small PLA are obtained externally by using two input pads for phase1 and phase2. Six pads of the chip are left unconnected. The pin assignments are illustrated in Figure 4.16. Figure 4.17 shows the layout of the final chip.

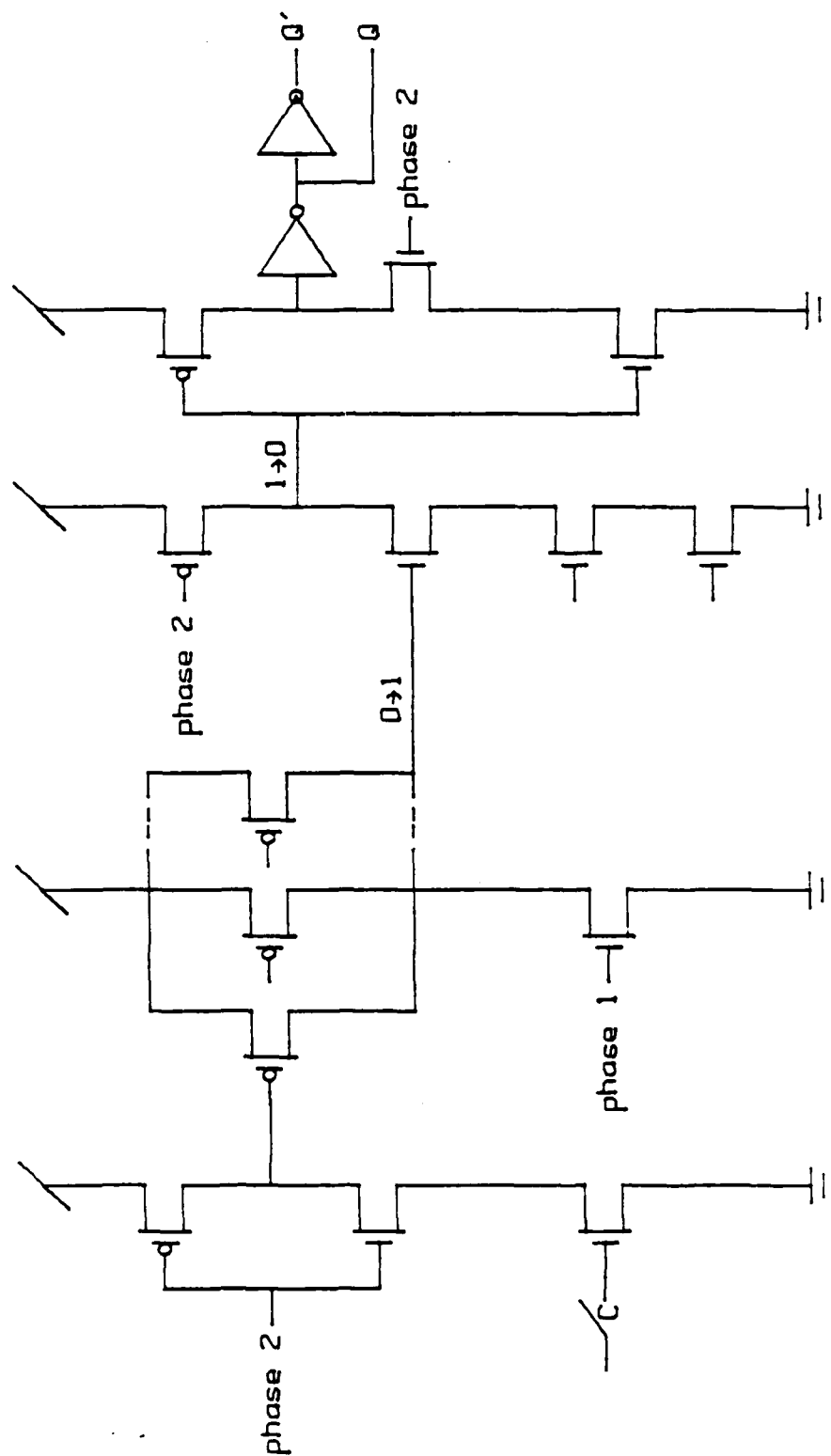


Figure 4.13 Circuit schematic of small ripple PLA

#### 4.2.1 Circuit Description

The AND-OR for this PLA is generated using NAND-NAND logic, where parallel PMOS transistors are used for the first NAND gate, and series NMOS transistors are used for the second NAND gate. This circuit concept was selected because it is convenient for PPL, the details of which are explained in the next chapter. The same circuit of PPL is chosen for this PLA so that both these circuits can be compared in area and performance. Chapter VI discusses such comparisons.

The circuit of the small PLA is illustrated in Figure 4.13. When phase1 is high, inputs are latched into the input latches, and the rest of the PLA is in the standby state or precharge state. When phase2 (phase1') goes high, the logic is evaluated in ripple-through manner and stored in the output latch. A detailed analysis of this circuit is conducted in Chapter V. The timing diagram for this PLA is shown in Figure 4.14.

#### 4.2.2 The Design example

The traffic light controller of Mead and Conway (6), the truth table of which is shown in Table 4.3, is programmed into this PLA. It has six inputs and eight outputs. One of the inputs as well as one of the output is unused. Inputs I3 through I5 are the external inputs, and inputs I1, I2 are the feed-back inputs from O1, O2. Outputs

next section are placed in a single I/O frame. The description of this frame is given in Section 4.3.

PLA inputs								PLA outputs									
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1
x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0
x	x	1	0	1	0	x	x	0	1	0	0	0	0	1	1	0	0
x	x	1	0	0	1	x	x	0	1	0	0	0	0	1	1	0	0
x	x	1	0	0	0	x	1	0	1	0	0	0	0	0	1	1	0
x	x	1	0	0	0	x	0	0	0	1	0	0	0	1	1	0	0
x	x	0	1	1	1	x	1	1	0	0	0	0	0	1	1	0	0
x	x	0	1	1	1	x	0	0	1	0	0	0	0	1	1	0	0
x	x	0	1	1	0	x	1	0	0	1	0	0	0	0	1	1	0
x	x	0	1	0	1	1	x	0	1	0	0	0	0	1	1	0	0
x	x	0	1	0	1	0	x	1	0	0	0	0	0	1	1	0	0
x	x	0	1	0	0	1	x	0	1	0	0	0	0	0	1	1	0
x	x	0	1	0	0	0	x	0	0	1	0	0	0	1	1	0	0
x	x	0	0	1	1	x	1	0	1	0	0	0	1	1	0	0	0
x	x	0	0	1	1	x	0	0	1	0	0	0	1	1	0	0	0
x	x	0	0	1	0	x	x	0	0	0	1	0	0	1	1	0	0
x	x	0	0	0	1	x	1	1	0	0	0	0	1	1	0	0	0
x	x	0	0	0	1	x	0	0	1	0	0	0	0	1	1	0	0
x	x	0	0	0	0	x	x	0	1	0	0	1	1	1	0	0	0

Table 4.2 Moderate PLA example

#### 4.2 Small PLA

Close observation of the moderate PLA reveals that logic is evaluated in three stages, NAND-NOR-NOR, though in ripple-through fashion to give AND-OR. A two-stage AND-OR ripple PLA is then designed which can be used for implementing small finite-state machines.

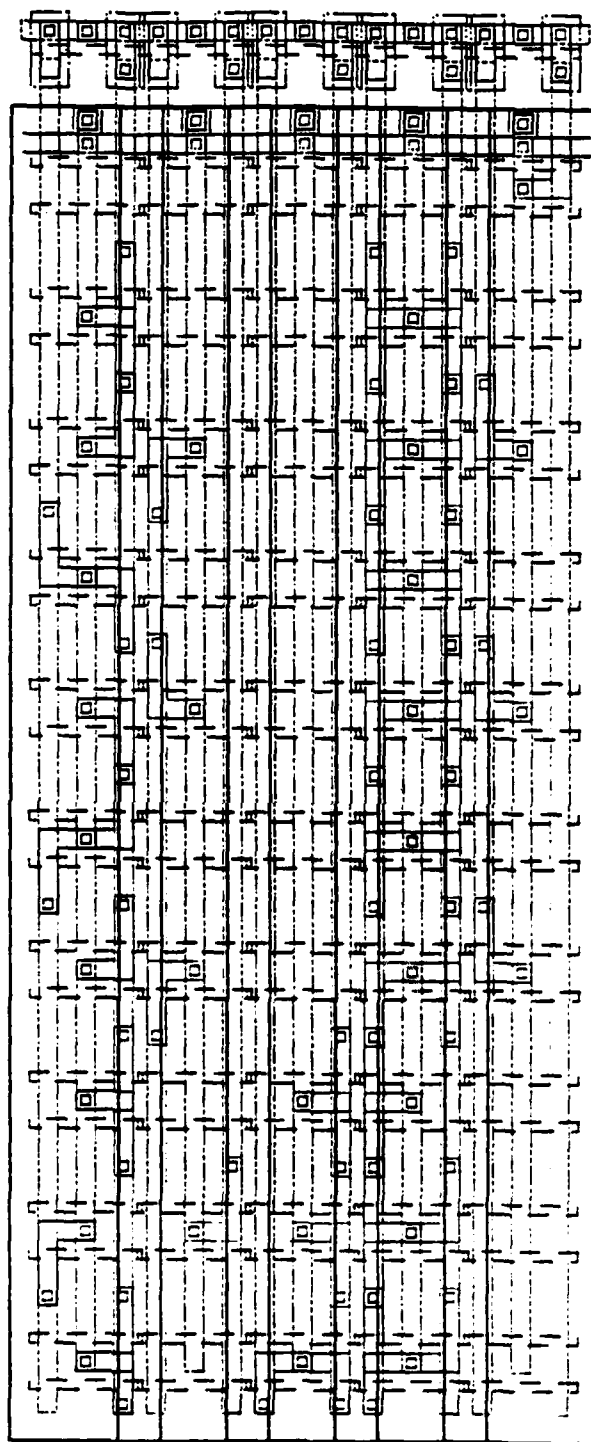


Figure 4.12 Layout of the OR plane for Moderate PLA



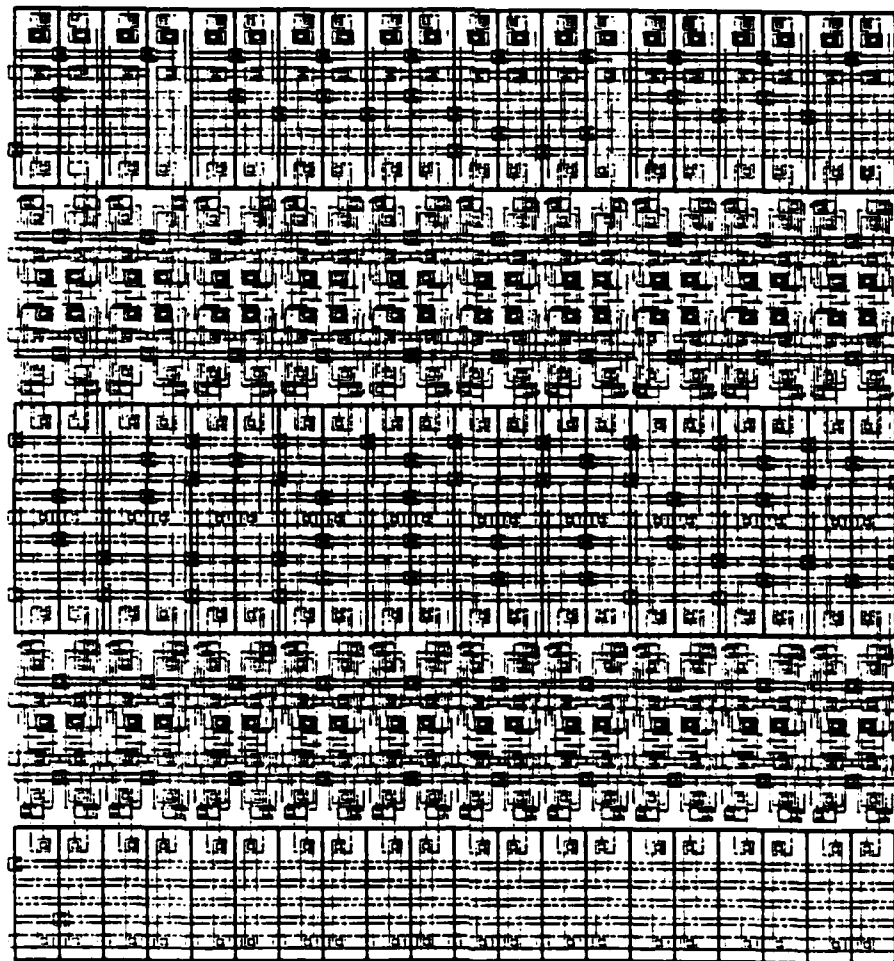


Figure 4.11 Layout of the AND plane for Moderate PLA

are used for the controller of Table 4.1. Output 010 is associated with the worst-case delay test.

BR Field				ZD	T	Select Address				C	Controls for SS			
B3	B2	B1	B0	Q/P		SA	SI	SF	SM		01F	02F	01B	02B
0	0	0	0	x	x	0	1	0	0	1	1	1	0	0
0	0	0	1	x	0	0	1	0	0	0	0	1	1	0
0	0	0	1	x	1	1	0	0	0	0	1	1	0	0
0	0	1	0	x	x	0	0	0	1	0	0	1	1	0
0	0	1	1	x	0	0	1	0	0	0	1	1	0	0
0	0	1	1	x	1	0	1	0	0	0	1	1	0	0
0	1	0	0	0	x	0	0	1	0	0	0	1	1	0
0	1	0	0	1	x	0	1	0	0	0	0	0	1	1
0	1	0	1	0	x	1	0	0	0	0	0	1	1	0
0	1	0	1	1	x	0	1	0	0	0	0	1	1	0
0	1	1	0	x	0	0	1	0	0	0	0	1	1	0
0	1	1	0	x	1	0	0	1	0	0	0	0	1	1
0	1	1	1	x	0	0	1	0	0	0	0	1	1	0
0	1	1	1	x	1	1	0	0	0	0	0	1	1	0
1	0	0	0	x	0	0	0	1	0	0	0	1	1	0
1	0	0	0	x	1	0	1	0	0	0	0	0	1	1
1	0	0	1	x	x	0	1	0	0	0	0	1	1	0
1	0	1	0	x	x	0	1	0	0	0	0	1	1	0

Table 4.1 Controller for subroutine stack and multiplexer of microsequencer

The AND plane is programmed in poly because the input lines of the AND plane are run in first metal and the OR plane is programmed in active. The layouts of the AND plane and of the OR plane, with the example of Table 4.2 programmed in, are shown in Figure 4.11 and Figure 4.12, respectively. This PLA along with the small PLA of the

The timing associated with the moderate PLA is shown in Figure 4.10. The cells of the moderate PLA are documented in Appendix C with their circuit schematics and layouts. The design example of the next section uses this library.

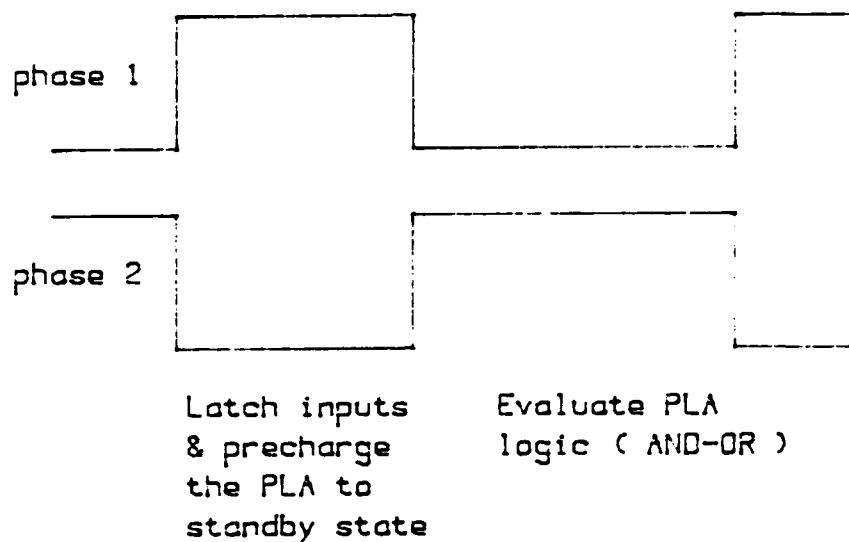


Figure 4 .10 Timing diagram of Moderate PLA

#### 4.1.1 The Moderate PLA Design example

The moderate PLA design has eight inputs and ten outputs. The example loaded is a controller for the multiplexer and the subroutine stack of a microsequencer (9). Table 4.1 shows the truth table of this controller. Table 4.2 is derived by two functions: the controller example described above and a worst-case delay test. A worst-case delay path is created by giving logic "1" to all the inputs. Inputs I1 through I6 and outputs O1 through O9

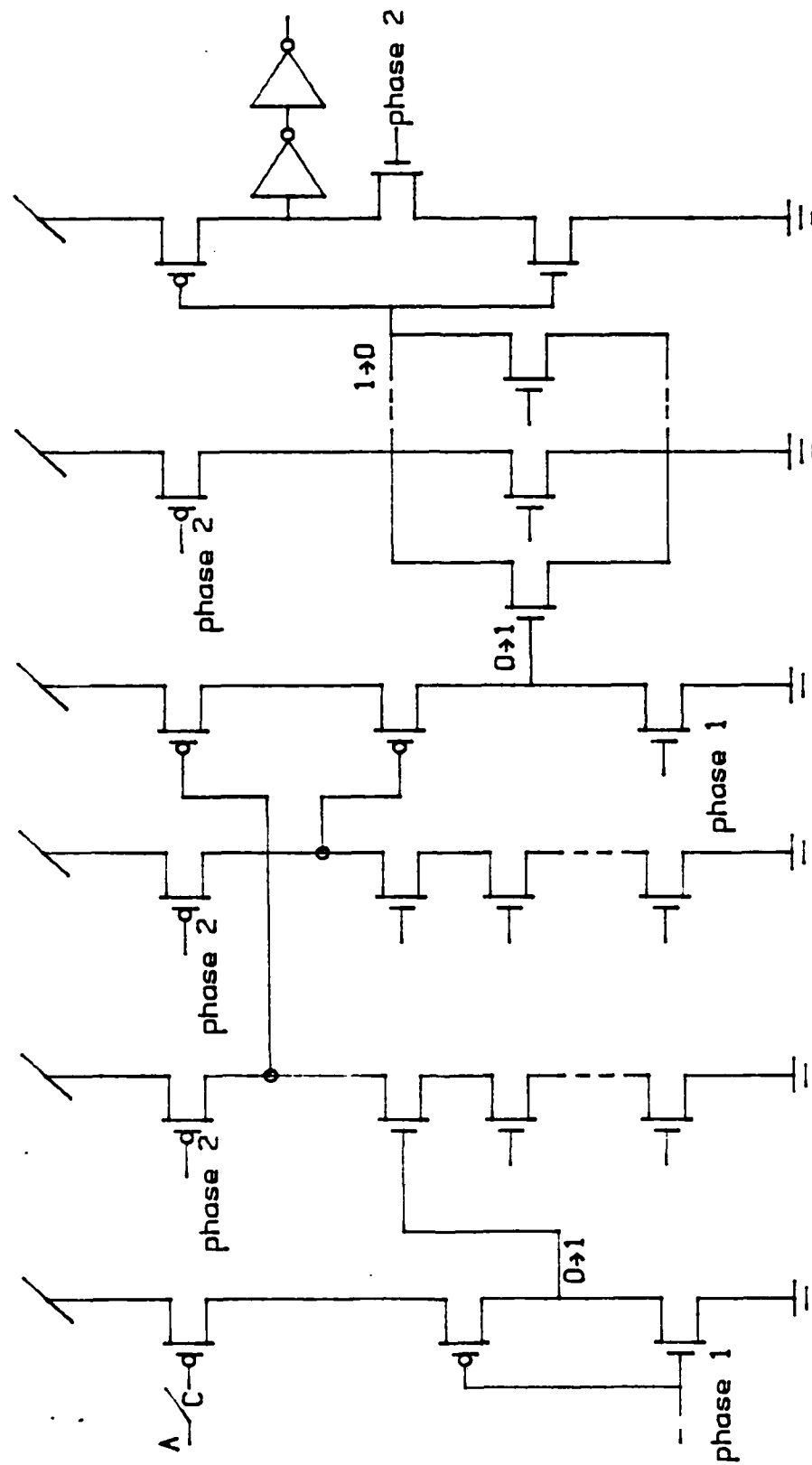


Figure 4.9 Overall circuit of Moderate PLA

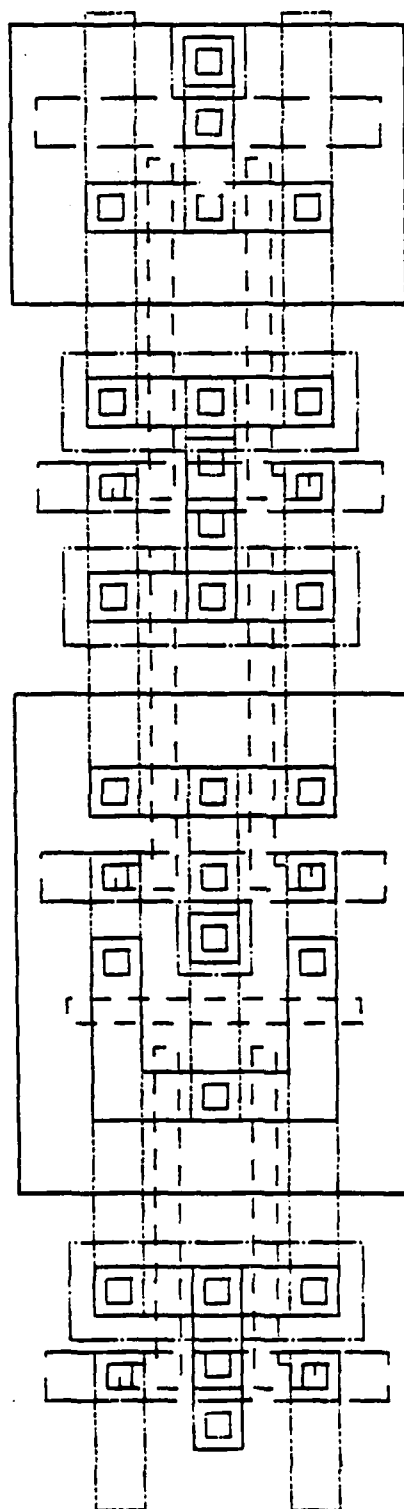


Figure 4.8 Output latch for Moderate PLA

In either case, a valid output is obtained during evaluation. The layout of the output latch with two inputs is shown in Figure 4.8.

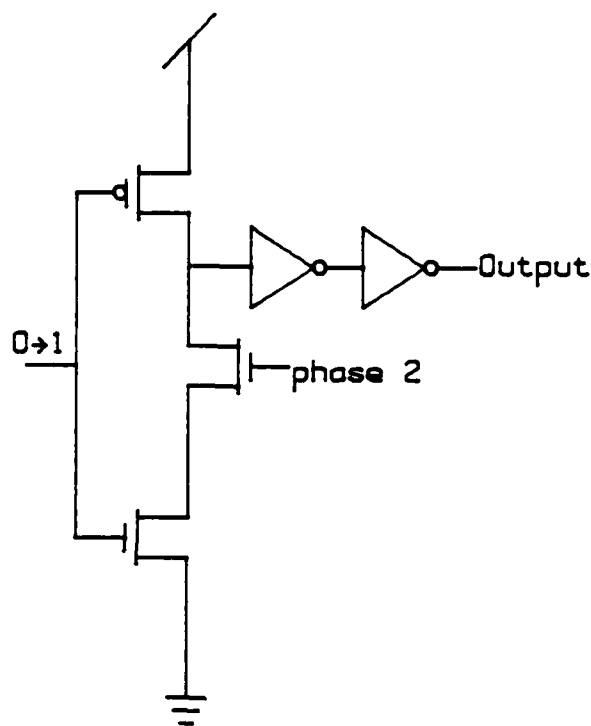
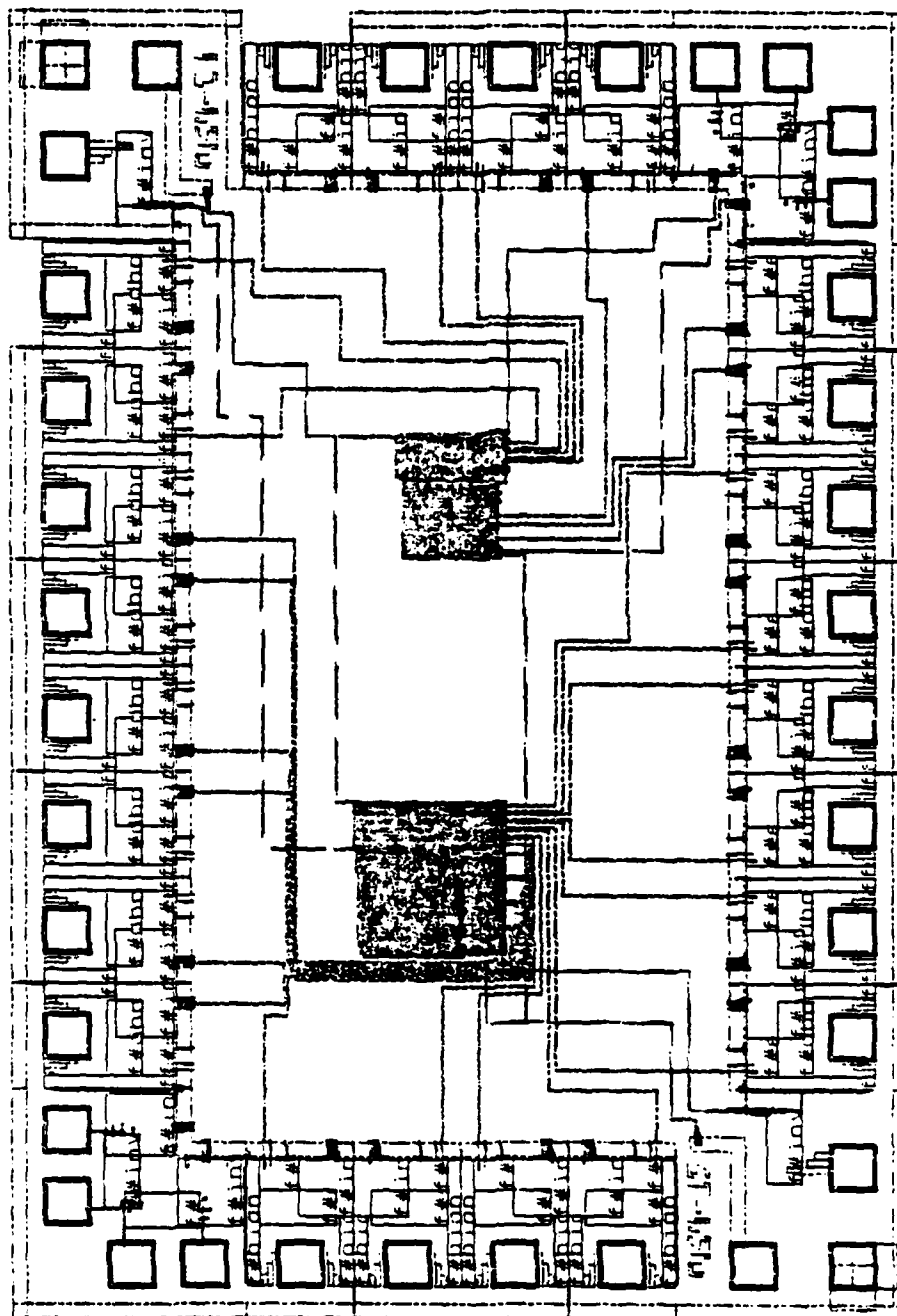


Figure 4.7 Dynamic NMOSTLY latch

Figure 4.9 illustrates the overall circuit of the moderate PLA. A two-phase clock scheme is used. During phase1, the external inputs are latched and stored on the gate capacitance of the input latch, and the rest of the PLA is in the standby state with inputs and outputs precharged to their respective states. For evaluation, phase2 goes high and the logic is evaluated in a ripple-through manner and stored in the output latches.



4.17 The final chip

## CHAPTER V

### PATH PROGRAMMABLE LOGIC

#### 5.1 Introduction

Path Programmable Logic (PPL) is a variation of PLA in which the AND and the OR planes are folded on top of each other. The logical OR is achieved by combining data along a single column, and the logical AND is achieved by combining data along a single row. All columns and rows can be broken at arbitrary locations and with memory elements distributed in the array, the PPL has the potential to have multiple independent, finite-state machines and data path modules on a single IC chip. This project aims at designing PPL in CMOS technology. The circuit is represented symbolically, and a cell library is developed to help in designing the layouts.

#### 5.2 PPL Circuit

Ripple circuitry is desirable for PPL to allow multiple levels of logic, for example, the AND-OR output can serve as another input into the AND plane.

The AND-OR for the PPL can be implemented in ripple-through fashion using both NOR-NOR and NAND-NAND logic. When using NAND and NOR, an inverter is needed between them to generate AND-OR; this involves additional



circuitry. The NOR-NOR can be implemented either with parallel N-gates driving series P-gates (Figure 5.1a), or with series P-gates driving parallel N-gates (Figure 5.1b). AND-OR with NOR-NOR requires series P-gates which are slower; therefore, it will not be discussed further. The NAND-NAND can also be implemented either with parallel P-gates driving series N-gates (Figure 5.2a) or with series N-gates driving parallel P-gates (Figure 5.2b). In both ways of implementing NAND-NAND, we have series N-gates either in the AND plane or in the OR plane. Minimizing the fan-in of the N-gate is very desirable. However, it is difficult to judge whether there will be a greater number of inputs to the AND plane or a greater number of implicants affecting one output in the OR plane, since this depends on the particular application. A detailed discussion of this concept is given in Chapter VI. From the layout point of view, the circuit with parallel P-gates driving series N-gates is used for this project. It should be noted that PPL can also be implemented with the remaining combinations.

In Figure 5.2a, which shows the circuit used for implementing the AND-OR, the AND plane outputs are precharged low and conditionally charged high. Its inputs are precharged high, and any one of the inputs going low will pull the AND output high. Similarly, the OR plane outputs are precharged high and conditionally discharged

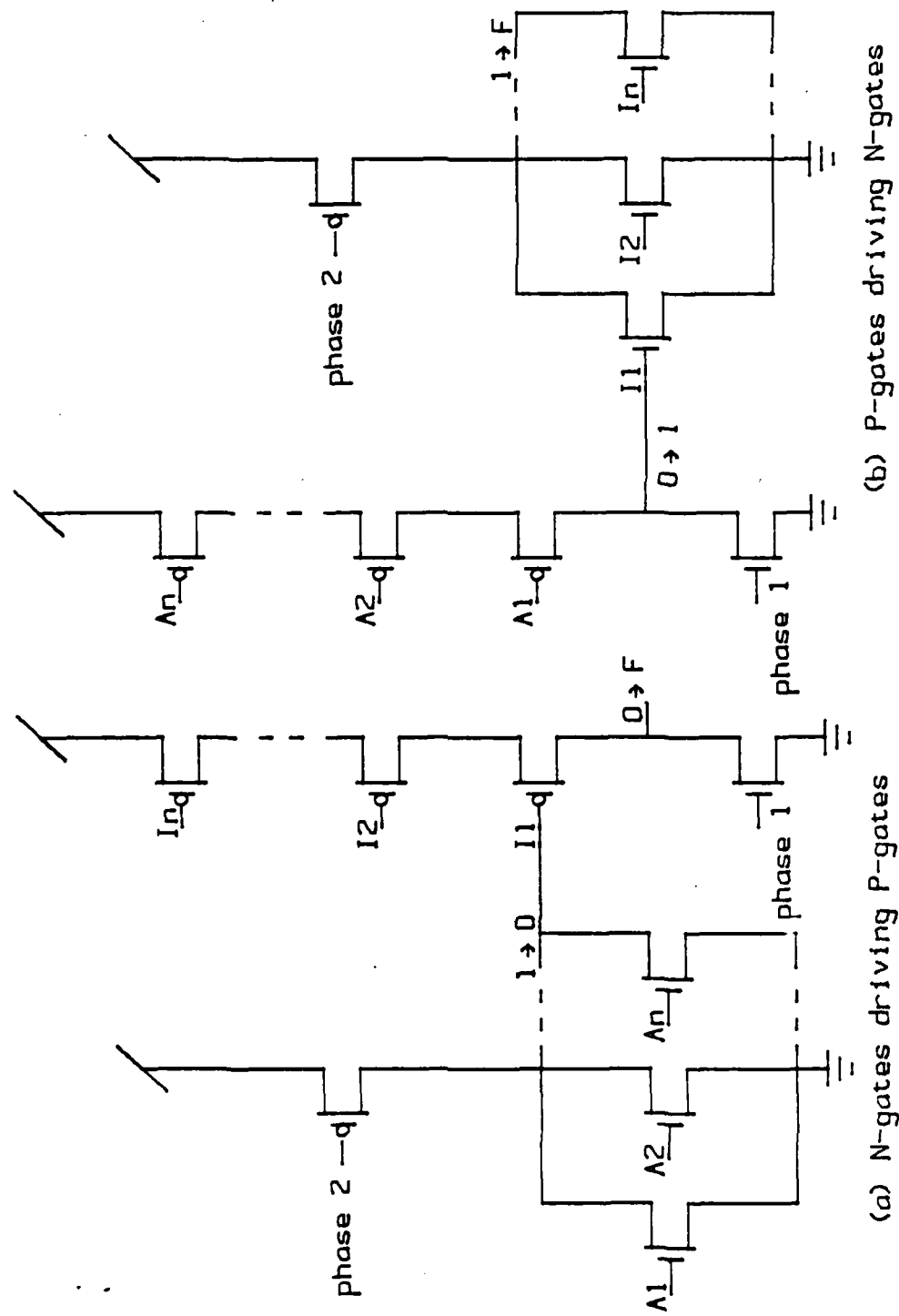
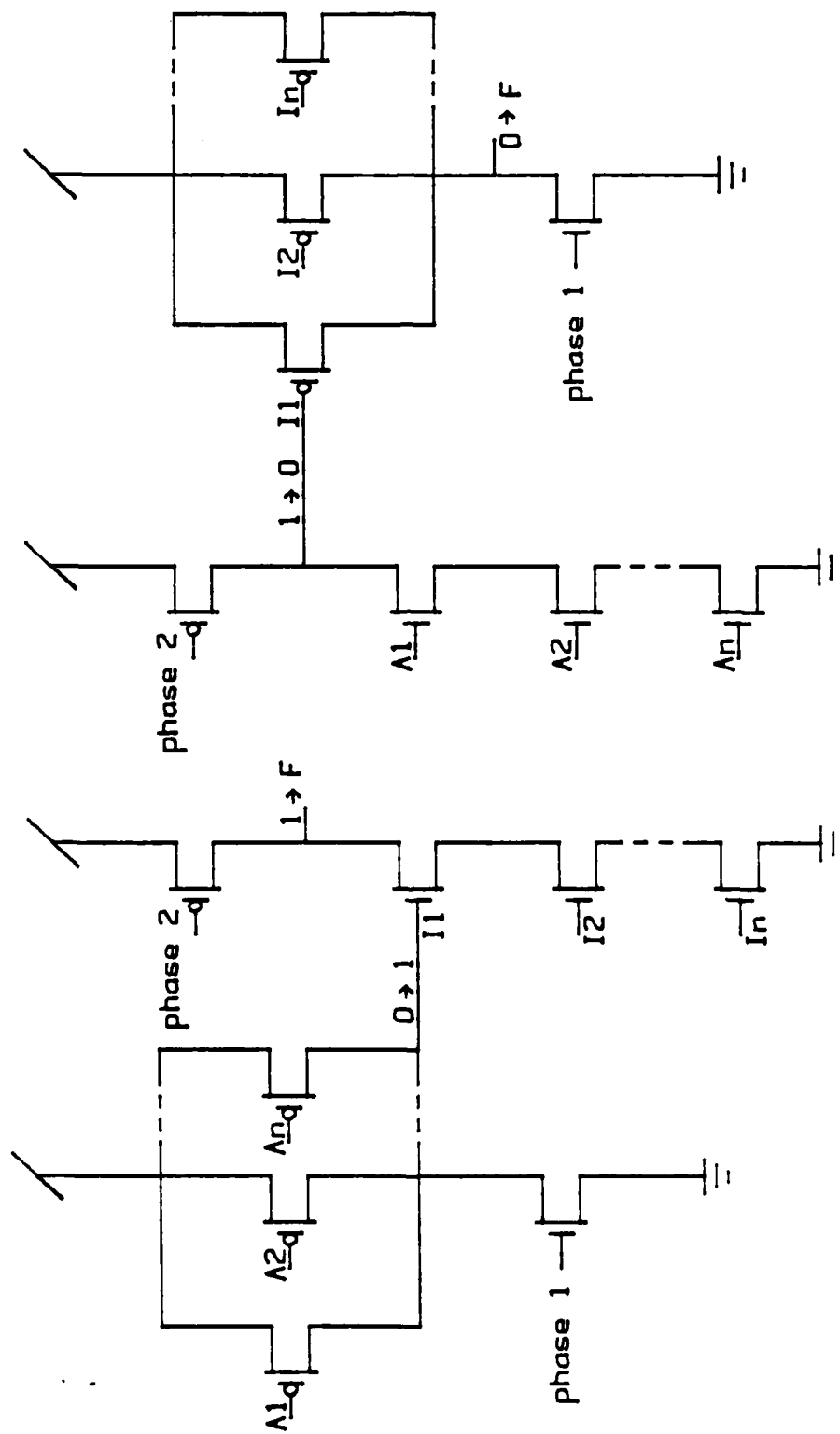


Figure 5.1 NOR-NOR implementation



(b) N-gates driving P-gates

(a) P-gates driving N-gates

Figure 5.2 NAND-NAND Implementation

low. For the OR output to discharge low, all of its inputs should be high; otherwise the output remains high. The point of concern here is charge splitting (discussed in Section 2.2 of Chapter II), which might result in the output falling to unacceptable levels.

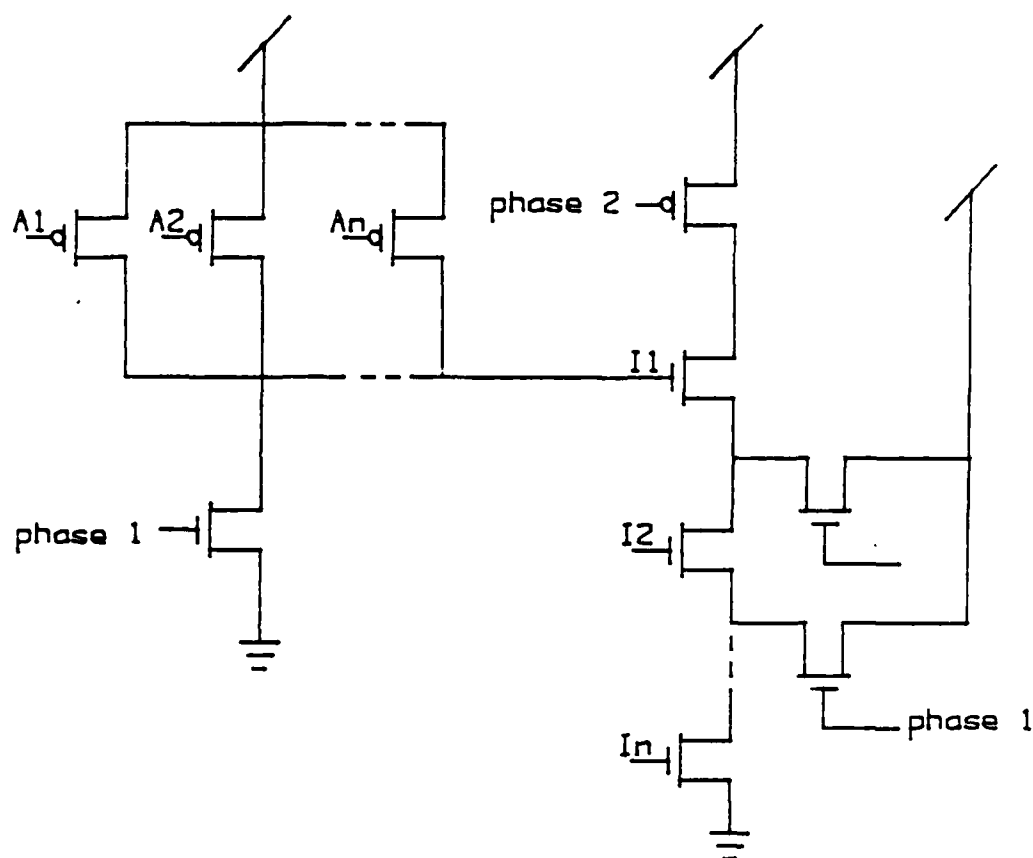


Figure 5.3 Improved circuit for AND-OR to avoid charge splitting

To overcome this, the improvised circuit shown in Figure 5.3 is used. Here the drain nodes of all the NMOS transistors, along with the output node, are precharged

high during precharge so that charge splitting is avoided during evaluation.

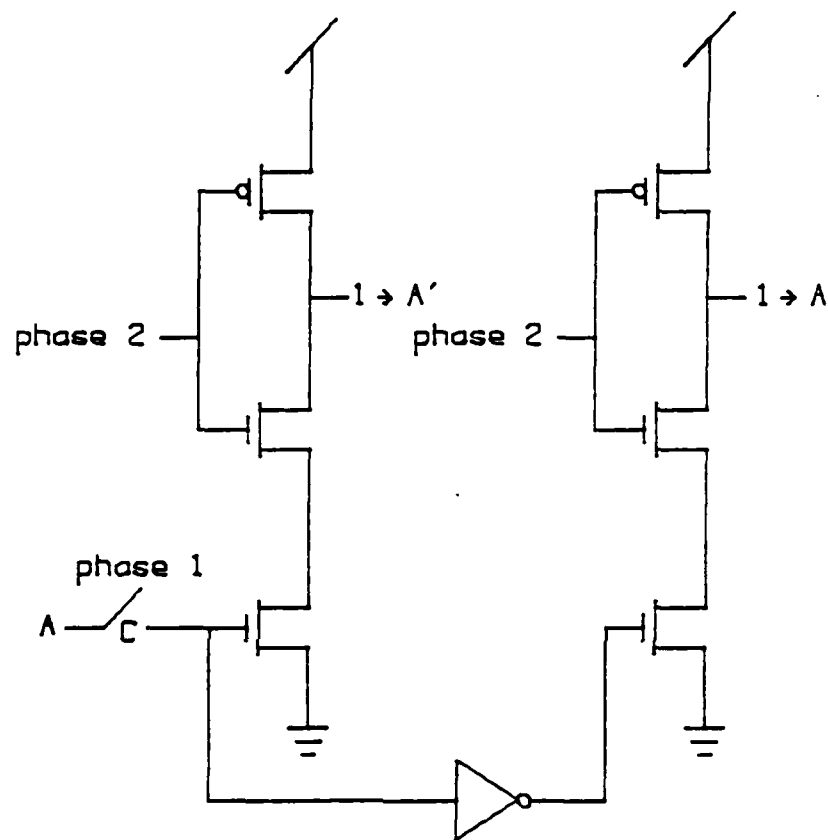


Figure 5.4 Input latch for PPL

The circuit schematic for the input latch is shown in Figure 5.4. During phase 1, the external inputs are latched into the circuit as the C switch is closed. The output is precharged high and conditionally discharged low, in accordance with the AND-OR input requirements. The input-partitioning discussed in Chapter III can be used

here as well. For simplicity, however, it is not considered in this design.

The dynamic Nmostly latch of Figure 5.5 is used as the output latch. The operation of this latch has already been discussed in Chapter IV. Many designers of state machines find set-reset latches useful. Therefore, a set-reset latch has been generated using classic NAND gates, as is illustrated in Figure 5.6. The NAND gates are used because the OR plane is precharged high. A high-going-low signal from the OR plane will set or reset the output. The timing diagram for the PPL is illustrated in Figure 5.7.

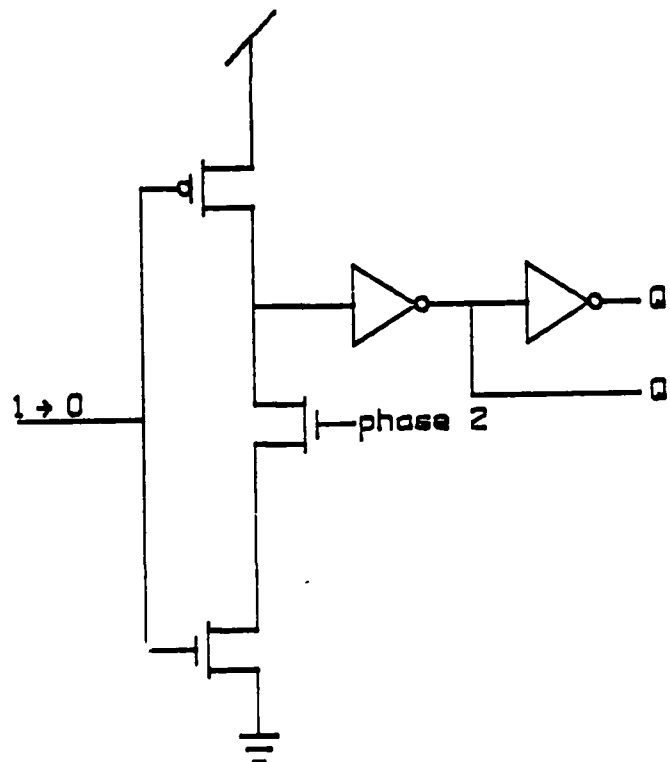


Figure 5.5 N-mostly latch

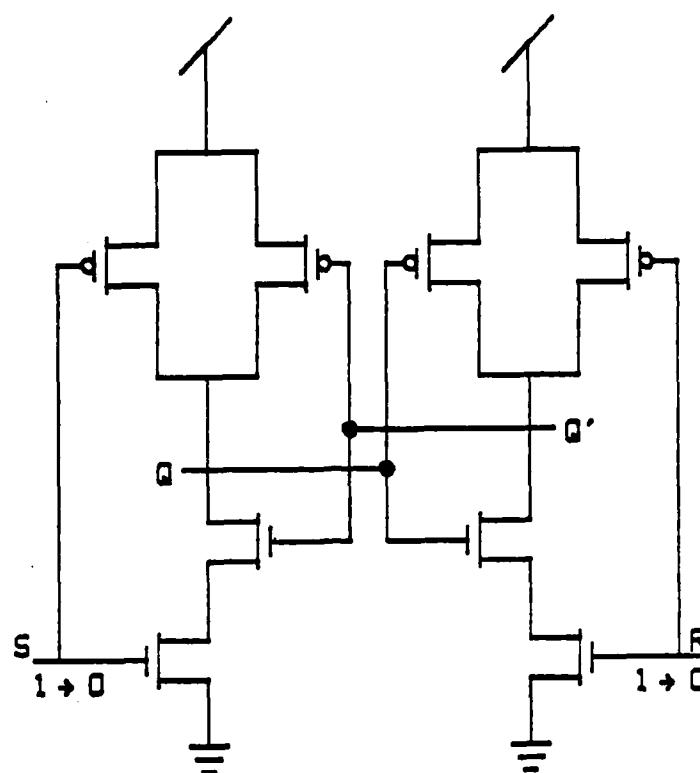


Figure 5.6 Set-Reset latch

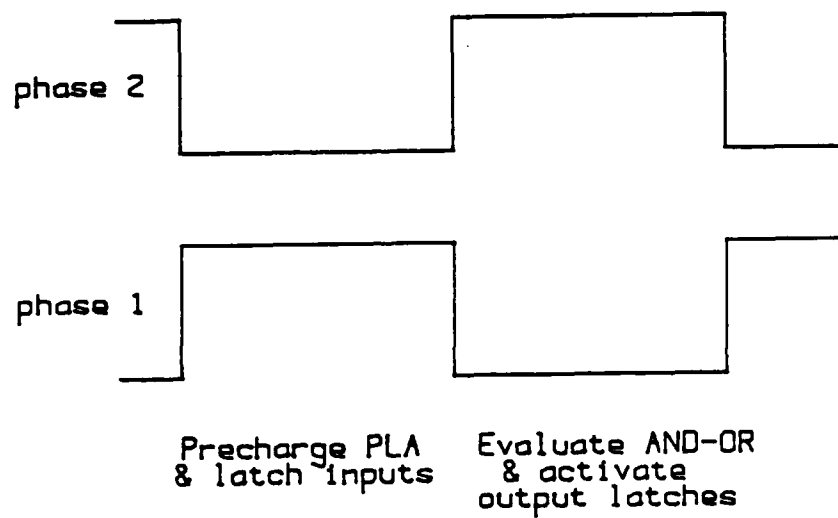


Figure 5.7 Timing diagram of PPL

### 5.3 PPL Cell Set

A precise PPL cell set is developed with the above concepts in view. For the present, this cell set can be used to implement finite-state machines. As mentioned previously, lambda-based scalable design rules documented in Appendix B are used for the design of these cells. The size of PPL cells is estimated in units; one unit is equal to eight by eight lambda. The reference point of all the cells is at the bottom left corner. Provision has been made to break the rows when designing.

The PPL cells are mainly divided into six categories: input cells, output cells, AND cells, OR cells, precharge cells, and interconnect cells. The first five of these are used for the PPL design in symbolic form, and the remaining type is used by the designer for interconnects. The AND and OR cells are considered to be the heart of the symbolic program. Figure 5.8 illustrates the floor plans of AND and OR cells in general. It shows that the inputs and outputs run vertically and that the implicants run horizontally. Therefore, a row line can be broken to have multiple implicants on a single row, and similarly a column line can be broken to have multiple inputs and outputs on a single column. This feature will be useful in minimizing the circuit area and in undertaking more levels of AND-OR.

The cells in the PPL cell set are:



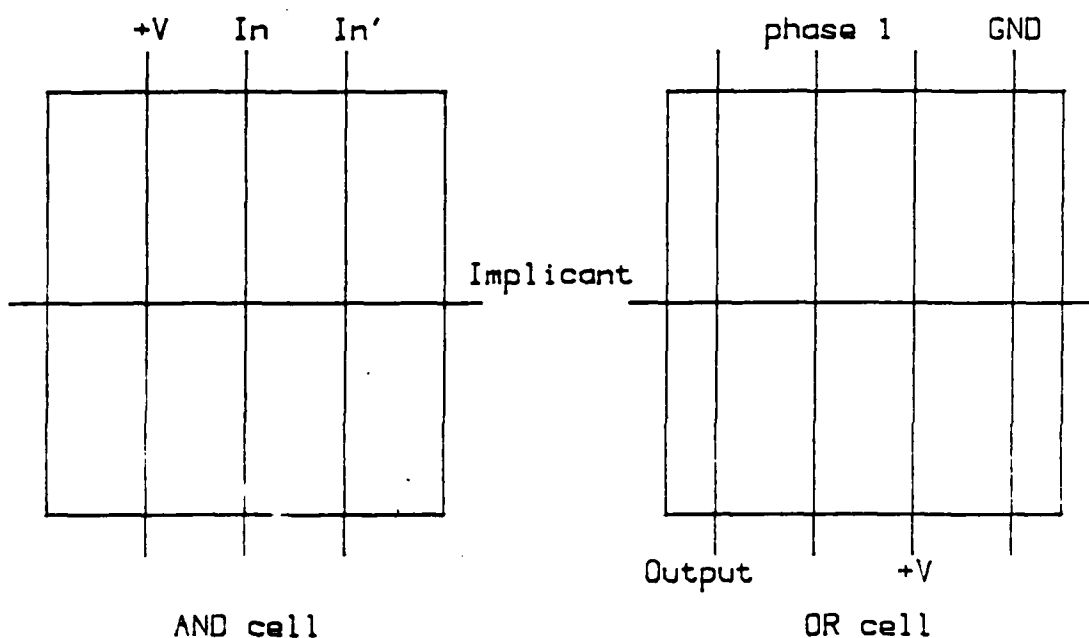


Figure 5.8 Floor plan of AND & OR cells

A) 0 (zero) : The combinational element of the AND plane which detects the false state of the input. Choosing the "0" cell in a column implies the selection of the complement signal in forming the implicant term associated with that respective row. The size of this cell is 2 by 2 units.

B) 1 (one) : The combinational element of the AND plane which detects the true state of the input. This cell is used if the true signal of the input is used in forming the implicant. The size of this cell is 2 by 2 units.

C) + (or) : The cell transmits the row element on to the OR column. This cell is used to perform the OR function of the implicants. The size of this cell is 3 by 2 units.

D) S' (set') : The combinational element detects the condition for not setting the set-reset flip-flop. This cell is used to place the implicant, which corresponds to the above condition, on the OR column. The size of this cell is 4 by 2 units.

E) R' (reset') : The combinational element detects the condition for not resetting the set-reset flip-flop. This cell is used to place the implicant, which corresponds to the above condition, on the OR column. The size of this cell is 4 by 2 units.

F) S'R' (set'-reset') : The combinational element detects the condition for not setting and resetting the set-reset flip-flop. This cell places the implicant on the respective OR columns. The size of this cell is 4 by 2 units.

G) IL (input latch) : The gated input latch (discussed in Section 5.2) is used to latch the input. The size of this cell is 2.5 by 13 units.

H) OL (output latch) : The Nmostly latch or gated D-latch which can be used to drive the external logic and to store the "state" code. The size of this cell is 2.5 by 12 units.

I) FF (set-reset latch) : A set-reset flip-flop which can be used as a memory element or to generate the state outputs. The size of this cell is 4 by 12.

J) PR (precharge-row) : An NMOS transistor used to precharge a row. The size of this cell is 1 by 1 units.

K) PC (precharge-column) : A PMOS transistor used to precharge a column. The size of this cell is 1 by 1 units.

L) RI (row-interconnect) : The cell used to interconnect a row. The size of this cell is 2 by 2 units.

M) CI (column-interconnect) : The cell used to interconnect a column. The size of this cell is 3 by 2 units.

N) SRI (set-reset column-interconnect) : The cell is used to interconnect set-reset column. The size of this cell is 4 by 2 units.

Cells A through K facilitate the design of a PPL in symbolic form; however, the layout designer needs additional cells L through N for interconnects. Row breaks are provided for 3 cells: zero, one, and RI. The rotated and mirrored versions of the PPL cell set are also available. Section 5.6 demonstrates the use of this PPL library with an example.

#### 5.4 Symbolic Representation

This section illustrates the use of symbols to represent the PPL designs. An Exclusive-OR example is used in this illustration. The logic diagram and its PPL symbolic representation are shown in Figure 5.9.

There are 3 rows and 3 columns in the program. The input latches occupy columns 1,2 of row 1 for latching inputs A and B. The output latch for F is placed at row 1, column 3. A "0" from input A is detected by the "0" placed in row 2, column 1, and a "1" from input B is detected by the "1" placed in row 2, column 2. The "0" and "1" in row 2 specify the AND condition, which is transmitted to the OR column by placing a "+" in row 2, column 3. The structure of row 3 can be analyzed using a similar argument. The OR condition is specified by the "+"s in column 3, rows 2 and 3.

## CHAPTER VI

CONCLUSIONS

In this concluding chapter, the different PLA and PPL design methodologies described in the preceeding chapters are compared. A discussion of future developments is also made.

In Chapter III, a "large" PLA design was described. The parallel gates structure of the AND and OR planes used for this PLA is more suitable when designing a controller for large systems where fan-in is high, as, for example, in the controller of a data path. The core cells of the large PLA have an area of 19 by 19 lambda in the AND plane, and 22 by 19 lambda in the OR plane. The PASCAL program of Appendix A facilitates programming the AND and OR planes of the PLA. This program is intended as only a temporary means of programming, however. In the long run, a more general approach for PLA generation, e.g., a Silicon Compiler, is needed.

The ripple PLA's described in Chapter IV are not useful for systems with large numbers of inputs/implicants; they can be used, however, for systems with few number of inputs/implicants. With a restriction in fan-in, the ripple PLA's are simple and faster and can drive successive

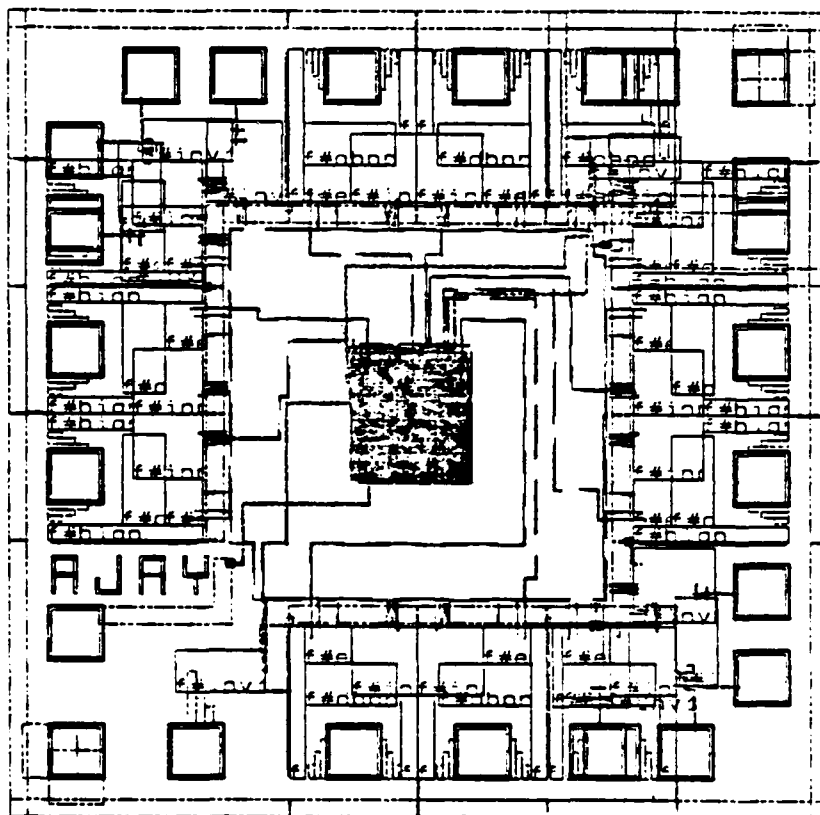


Figure 5.16 The PPL chip

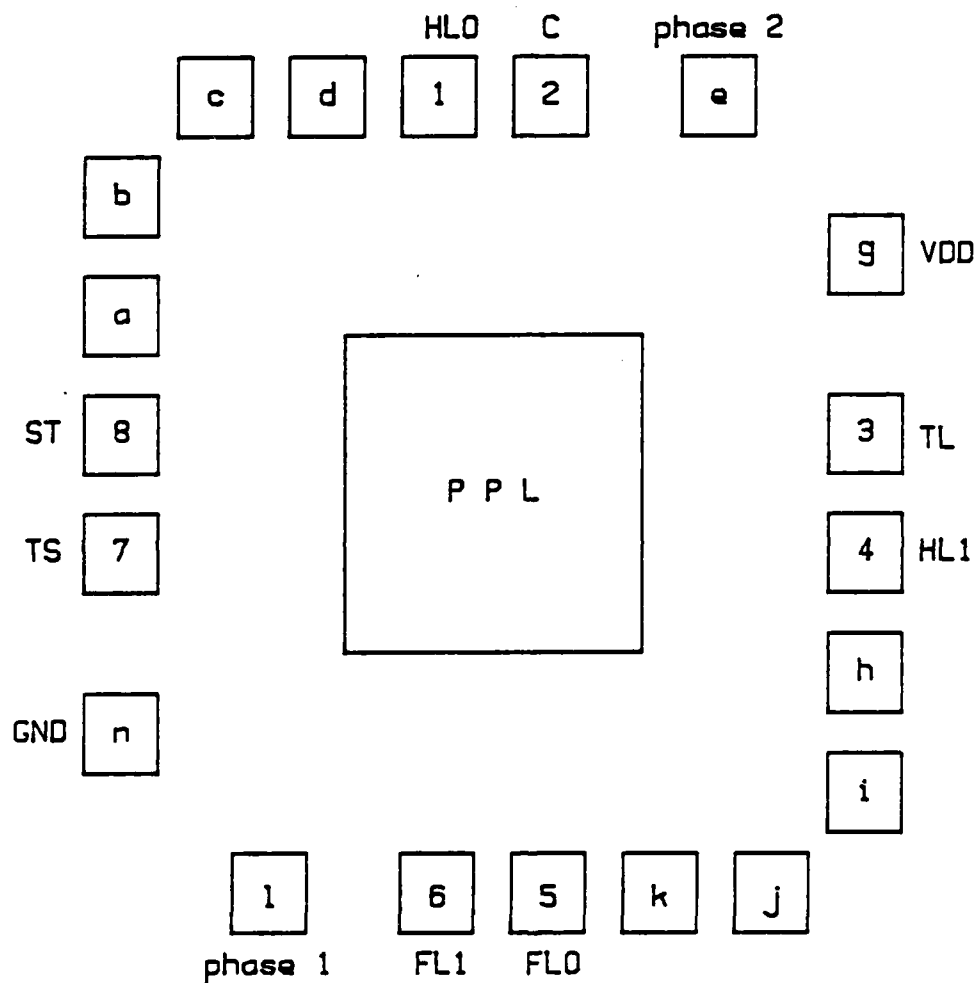


Figure 5 .15 Pin configuration of PPL chip

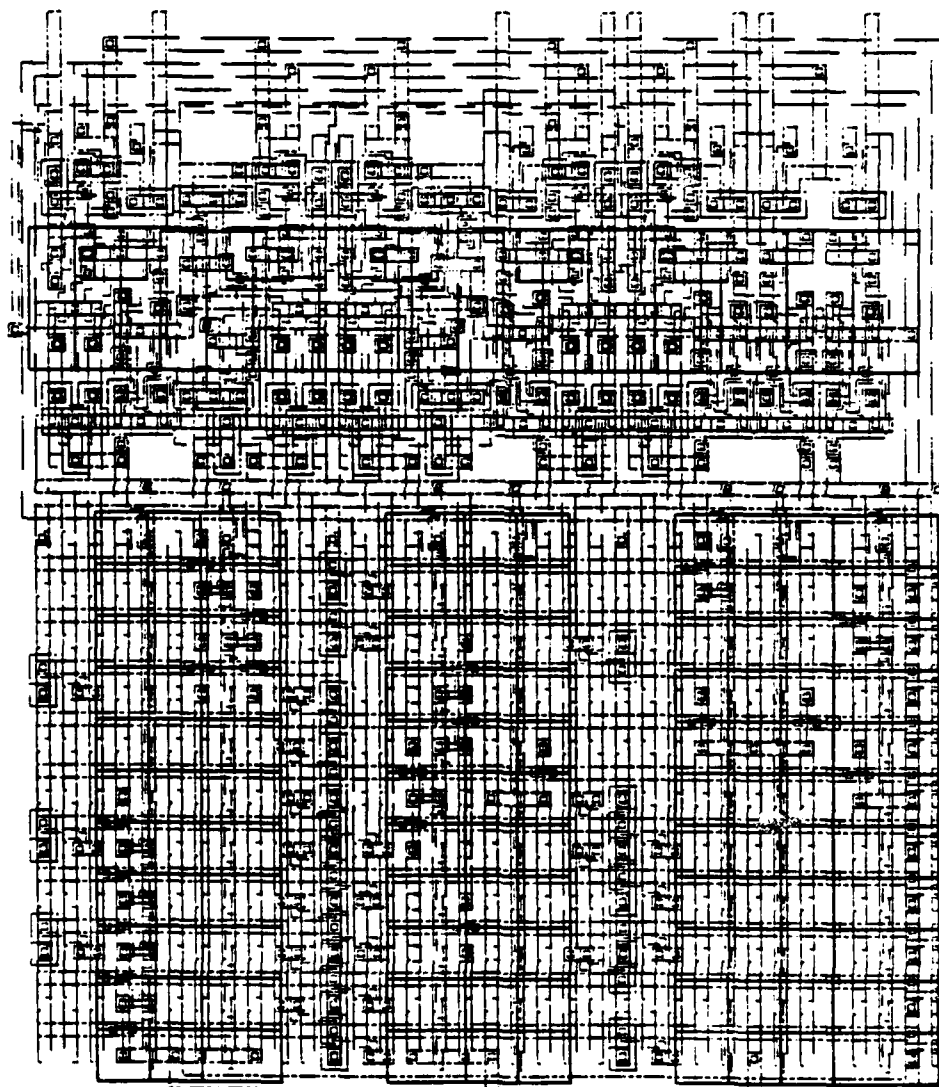


Figure 5.14 The PPL layout



IL	OL	FFm	ILm	IL	FF	OLm	ILm	IL	OL	OLm	OL
RI	CI	S'm	RIm	0	SRI	CIm	RIm	RI	+	CIm	CI PR
RI	CI	R'm	RIm	1	SRI	CIm	RIm	RI	CI	CIm	+
RI	CI	S'R'm	RImb	Rlb	S'	CIm	Om	RI	CI	CIm	CI PR
RI	CI	SRI m	1m	RI	S'	CIm	RIm	RI	+	+m	CI PR
RI	CI	SRI m	Om	RI	R'	+m	RIm	RI	CI	CIm	+
0	+	SRI m	1m	Ob	R'	CIm	1m	0	CI	CIm	CI PR
RI	+	SRI m	1m	1	SRI	CIm	1m	0	CI	CIm	CI PR
RI	+	SRI m	Om	1b	S'	CIm	RIm	0	CI	CIm	CI PR
0	+	SRI m	Om	0	SRI	CIm	Om	RI	CI	CIm	CI PR
RI	+	SRI m	Om	0	SRI	CIm	RIm	0	CI	CIm	CI PR

Figure 5.13 Work sheet for chromatics

The layout of the TLC has been designed using the symbolic program generated. Since suitable software for automatic generation of the layout from the symbolic program has not been developed yet, the layout in this example has been done manually on a Chromatics graphic computer. The Chromatics user creates a work-sheet from the symbolic program as shown in Figure 5.13. An "m" following the PPL cell denotes that the cell is mirrored and a "b" denotes a row break. The placing of the cells can be better understood if one goes through the PPL cell set thoroughly and inspects the work-sheet in Figure 5.13. The layout of the TLC generated with the aid of the work-sheet is shown in Figure 5.14.

A 20-pad frame has been used for this PPL example. The pin configurations are shown in Figure 5.15. Eight pins are not used. The picture of the final chip is shown in Figure 5.16.

	TS	ST	Y0	Y1	HLO	C	TL	HL1	FL0	FL1	
	↓	↑	↓	↓	↑	↓	↓	↑	↑	↑	
	IL	OL	FF	IL	IL	OL	IL	IL	OL	OL	OL
R1			S'						+		
R3			R'		1						+
R2	0		S'R'	*			0				
R4											
R6				1					+		
R7				0		+					+
R8	0	+		1			1	0			
R10		+		1			1	0			
R11		+		0							
R5				1	*			0			
R12	0	+		0			0				
R13		+		0				0			

Figure 5.12 Minimized symbolic program of TLC

	C	TL	TS	Y0	Y1	ST	HLO	HL1	FLO	FL1	
	↓	↓	↓	↓	↓	↑	↑	↑	↑	↑	
	IL	IL	IL	IL	FF	IL	OL	OL	OL	OL	OL
R1					S'	0			+		
R2			0		S'R'						
R3					R'	1					+
R4	0										
R5		0			S'						
R6				1	S'				+		
R7				0	R'		+				+
R8	1	0			R'						
R9			0	1		0	+				
R10	1	0		1		1	+				
R11			0	0		1	+				
R12	0			0		0	+				
R13		0		0		0	+				

Figure 5.11 Symbolic program for TLC

"1" and "0" respectively. They are collected on the rows (ANDed terms) and the "S'", "R'", or "+" cell is activated accordingly. The row precharge cells are placed wherever it is possible in the array, and the column precharge cells are placed on the top of the column array. The symbolic program, with suitable latches placed, is shown in Figure 5.11. This can be minimized by proper arrangement of the inputs and outputs and by using row breaks and column breaks. The modified symbolic program is shown in Figure 5.12, where there are just 10 rows instead of 13 in Figure 5.11. A cross in the program indicates a row break.

Inputs					Outputs									
C	TL	TS	Present state		Next state				$\overline{ST}$	$\overline{HL0}$	$\overline{HL1}$	$\overline{FL0}$	$\overline{FL1}$	
			Yp0	Yp1	Yn0		Yn1							
					$\overline{S}$	$\overline{R}$	$\overline{S}$	$\overline{S}$						
x	x	x	x	0	1	0	0	0	0	0	1	0	0	
x	x	0	x	x	1	1	0	0	0	0	0	0	0	
x	x	x	x	1	0	1	0	0	0	0	0	0	1	
0	x	x	x	x	0	0	1	0	0	0	0	0	0	
x	0	x	x	x	0	0	1	0	0	0	0	0	0	
x	x	x	1	x	0	0	1	0	0	0	1	1	0	
x	x	x	0	x	0	0	0	1	0	1	0	0	1	
1	0	x	x	x	0	0	0	1	0	0	0	0	0	
x	x	0	1	0	0	0	0	0	1	0	0	0	0	
1	0	x	1	1	0	0	0	0	1	0	0	0	0	
x	x	0	0	1	0	0	0	0	1	0	0	0	0	
0	x	x	0	0	0	0	0	0	1	0	0	0	0	
x	0	x	0	0	0	0	0	0	1	0	0	0	0	

Table 5.3

shown in Table 5.3. The table indicates that the simplified version has few number of ORing terms, a highly desirable characteristic for this design; however, an increase in the number of implicants is observed.

<u>Inputs</u>					<u>Outputs</u>								
Present state					Next state								
C	TL	TS	Yp0	Yp1	$\overline{Yn0}$		$\overline{Yn1}$		$\overline{ST}$	$\overline{HL0}$	$\overline{HL1}$	$\overline{FL0}$	$\overline{FL1}$
					S	R	S	R					
0	x	x	0	0	1	x	1	x	1	1	1	0	1
x	0	x	0	0	1	x	1	x	1	1	1	0	1
1	1	x	0	0	1	x	0	1	0	1	1	0	1
x	x	0	0	1	1	x	x	1	1	1	0	0	1
x	x	1	0	1	0	1	x	1	0	1	0	0	1
1	0	x	1	1	x	1	x	1	1	0	1	1	1
0	x	x	1	1	x	1	1	0	0	0	1	1	1
x	1	x	1	1	x	1	1	0	0	0	1	1	1
x	x	0	1	0	x	1	1	x	1	0	1	1	0
x	x	1	1	0	1	0	1	x	0	0	1	1	0

Table 5.2

Once the truth table is generated, the next step is to generate the symbolic program. It is generated from the table by replacing the 1's in the set' and reset' outputs by "S'" and "R'" respectively and by putting a "+" in place of 1's in the rest of the outputs. All these cells are collected on the columns to drive output latches. The 1's and 0's of the inputs in the truth table are replaced by

terms, possibly with an increase in implicant terms. The example of Section 5.6 illustrates this simplification.

### 5.6 PPL Example

A traffic-light controller, TLC (6), has been designed using the PPL cell set and philosophy. Table 5.1 illustrates the state transition of TLC. For the state outputs, set-reset latches are used, and for the remaining outputs, Nmostly latches are used. All inputs are latched using the gated input latches.

<u>Inputs</u>					<u>Outputs</u>								
Present state					Next state								
C	TL	TS	Yp0	Yp1	Yn0		Yn1		ST	HL0	HL1	FL0	FL1
					S	R	S	R					
0	x	x	0	0	0	x	0	x	0	0	0	1	0
x	0	x	0	0	0	x	0	x	0	0	0	1	0
1	1	x	0	0	0	x	1	0	1	0	0	1	0
x	x	0	0	1	0	x	x	0	0	0	1	1	0
x	x	1	0	1	1	0	x	0	1	0	1	1	0
1	0	x	1	1	x	0	x	0	0	1	0	0	0
0	x	x	1	1	x	0	0	1	1	1	0	0	0
x	1	x	1	1	x	0	0	1	1	1	0	0	0
x	x	0	1	0	x	0	0	x	0	1	0	0	1
x	x	1	1	0	0	1	0	x	1	1	0	0	1

Table 5.1

Table 5.2, with set' and reset', is derived from Table 5.1. The next step is to generate a simplified version, as

in "0" state, and that this implicant driving one of the series "N" switches forces the overall gate to be OFF, thus forcing its output to remain in its precharged "1" state. Therefore, for a true (positive) logic there is no transition from "1" to "0" at the AND-OR output. This kind of output cannot drive set-reset latches since they anticipate a "1" going to "0" for set or reset.

This problem can be overcome in two ways:

1) By having series NMOS transistors driving parallel PMOS transistors. For the selected input combination, all the NMOS devices are ON, pulling the output node low, which in turn activates the parallel PMOS device. Thus a transition from the precharged "0" state to "1" is achieved. The core cell of this arrangement has dimensions of  $26$  by  $16$  lambda as opposed to  $16$  by  $16$  lambda as in the previous arrangement. Therefore it requires almost double the area.

2) By generating set' and reset' at the AND-OR output.

Since the former method of getting around the problem of set and reset is not economical, the latter method has been used. The design can also be simplified by using Karnaugh maps. With this technique a more simplified version is generated, often leading to a decrease in ORed



NAND; NMOS transistors in series generate the second NAND. This is illustrated with switches in Figure 5.10.

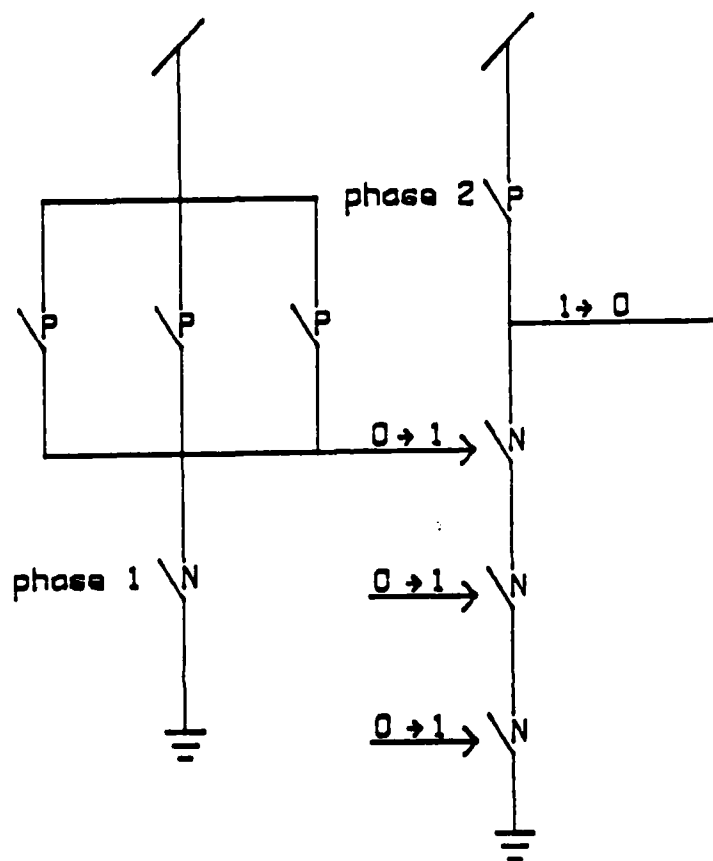


Figure 5.10 AND-OR functioning

For example, in the first stage, a combination of 101 opens all the "P" switches; then the output of this gate remains in the precharge state of "0". For all remaining combinations, at least one of the switches will be closed, pulling the output node to "1". In the second stage, the output goes to ground only if all the switches are closed by having their inputs high; otherwise the output remains in the precharged high state. It is of interest to note that the selected input combination will have its implicant

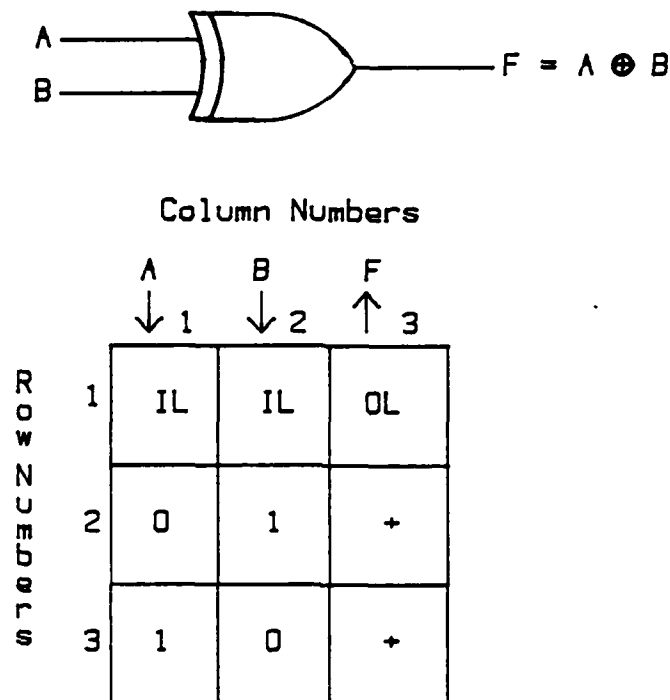


Figure 5.9 Symbolic representation of Exclusive-OR

For the conditions shown in the symbolic program, the output is "1"; otherwise it is "0", which implies an Exclusive-OR function. With this kind of symbolic program, the user need not worry about the internal circuitry. It is also easy to walk through the program and make necessary changes. A cell set which can be used for writing symbolic programs is developed in the next section.

### 5.5 Design Methodology

The AND-OR for the PPL is generated using NAND-NAND logic. PMOS transistors in parallel generate the first

stages during the same clock period. The "moderate" PLA is used as a compromise between the "large" and the "small" PLA's. For this PLA, the fan-in can be more than that of a small PLA, but with a sacrifice in circuit area and speed. It is advisable to use such a system if the fan-in needs to be increased, thus preserving the ripple-through nature. The core cells have an area of 36 by 28 lambda in the AND plane, and 23 by 28 lambda in the OR plane. The "small" PLA is suitable for implementing small finite-state machines. The core cells of this PLA have an area of 32 by 32 lambda in the AND plane, and 26 by 32 lambda in the OR plane. Table 6.1 lists the dimensions of the AND and OR planes of these PLA's.

Table 6.1 is not sufficient for the comparison of the different PLA's studied. In a PLA design, there are input latches, output latches, and precharge devices apart from the AND and OR plane circuits. This periphery, along with its interconnects, plays an equally important role in the layout design. For example, the OR plane dimensions of the large PLA are limited by the output latches, and similarly, the AND plane dimensions of the small PLA are limited by the OR plane. Therefore, a more reasonable comparison of the PLA discussed is to estimate the total area for generating a PLA. The total area for four inputs, eight implicants and eight outputs using the different PLA schemes is listed in Table 6.2. This table indicates that

the size of the large PLA is smaller than that of the ripple PLA's. Every PLA structure discussed has its own advantages. This project introduces a finite-state machine designer to the different PLA structures, so that the best PLA suitable for the application can be chosen.

The PPL design described in Chapter V is a folded form of the PLA. The circuit used for the PPL is similar to that of the small PLA circuit. Tables 6.1 and 6.2 illustrate the core cells area and the total area for implementing a PPL for four inputs, eight implicants and eight outputs, respectively. A symbolic form of representing the design is described, and a suitable cell set is developed for designing the PPL. The design example in Chapter V also gives an insight into the procedure of designing the PPL. In order to activate the set-reset latches, set' and reset' are required instead of set and reset outputs from the OR plane. This increases the number of series NMOS transistors. To overcome this feature, minimization techniques are applied to decrease the series transistor count. By this procedure, a penalty is paid in the number of rows (implicants). In the example of Section 5.6, the number of rows increased from 10 to 13. This concept of the PPL needs further development. A possible solution is to use D-latches instead of SR-latches.

The direct comparison of the PPL with the PLA yields four immediate advantages:

1) The area of the PPL is 56,472 square lambda against the 60,300 square lambda of the small PLA.

2) The row breaks and column breaks allow more implicants on a single row and more outputs on a single column. In the PLA, rows and columns are dedicated for single implicants and outputs.

3) In the PPL, the inputs and outputs are alternated where suitable. This is possible because of the folded nature of the AND-OR planes. This capability of the PPL is helpful for a compact external routing, whereas the PLA lacks the capability.

4) The PPL can be expanded to handle functions like  $(AB+CD)*(A'C+AC'D')+(EF+A'D)*(AEF'+CDE')$  or  $(ABC+A'EF)*B'CD$ , which are difficult to handle using the PLA. With this capability, the PPL can be used for realizing more generalized functions than the conventional 2-level AND-OR functions handled by the PLA.

Therefore, the PPL is definitely a more powerful tool than the PLA. The PPL can also be made useful for implementing functions other than finite-state machines. Addition of storage cells at regular intervals in the array makes the PPL convenient for data path modules as well. For implementing functions using the PPL, the symbolic program will be more convenient for the designer. This,

with the aid of suitable software, can greatly reduce the design effort. The PPL design methodology with software support has the potential to rival semi-custom designs in the design effort and to provide densities more nearly approaching those of full-custom designs. These points should be considered in the future development of the PPL.

	Large PLA	Moderate PLA	Small PLA	PPL
Dimensions of (in $\lambda$ )				
AND cell	: 19 x 19	36 x 28	32 x 32	32 x 32
OR cell	: 22 x 19	23 x 28	26 x 32	48 x 32

Table 6.1 Comparison of AND-OR planes of different PLA's

	Area in $\lambda$ square
Large PLA	: 57,928
Moderate PLA	: 61,780
Small PLA	: 60,300
PPL	: 56,472

Table 6.2 Circuit area for 4-inputs & 8-outputs using  
different PLA's

APPENDIX A

PASCAL PROGRAM



PLA PROGRAM  
By  
NAINI

Input: The input to the program is the truth table of the function to be programmed into the PLA. The user also need to select the input partitioning option.

Output: The program generates a MSL (Mississippi State Layout Language) file which places transistors in the AND plane, OR plane and the input circuit of the PLA.

PROGRAM Main (Input, Output, Andarray, Orarray) ;

CONST

```
AC1 = 'ac1' ;
AC2 = 'ac2' ;
AC3 = 'ac3' ;
AC4 = 'ac4' ;
OC1 = 'oc1' ;
OC2 = 'oc2' ;
OC3 = 'oc3' ;
OC4 = 'oc4' ;
OR_ROW = 28 ;
OR_COL = 28 ;
AND_ROW = 28 ;
AND_COL = 64 ;
```

TYPE

```
Str5 = Packed Array [ 1 .. 5 ] of Char ;
Str4 = Packed Array [ 1 .. 4 ] of Char ;
Str3 = Packed Array [ 1 .. 3 ] of Char ;
N_Type = Array [ 1 .. 64 ] of Str4 ;
PLA_rec = Record
    Row,
    Column      : Real ;
    Cell        : Str3 ;
End ;
And_Type = Array [ 1 .. AND_ROW,
                  1 .. AND_COL ] of PLA_rec ;
Or_Type  = Array [ 1 .. OR_ROW,
                  1 .. OR_COL ] of PLA_rec ;
```

VAR

```
x,
y
And_Plane      : Integer ;
Or_Plane       : And_Type ;
N_cell         : Or_Type ;
Ch             : N_Type ;
i,
j,
AA_Output      : Integer ;
```

```

A_Input,
B_Input      : Char      ;
Andarray,
Orarray      : Text      ;

PROCEDURE Validationcell (VAR N_cell : N_Type ;
                          VAR Nor_in,
                          a,
                          b,
                          c : Str5 ;
                          one,
                          two,
                          three : Str4 ;
                          x : Integer ) ;

BEGIN
  If Nor_in = a Then
    N_cell[x] := one
  Else
    If Nor_in = b Then
      N_cell[x] := two
    Else
      If Nor_in = c Then
        N_cell[x] := three ;
      Writeln ('c ', N_cell[x] : 4, ', ',
        (-615 + ((x-1) Div 4) * 38) : 0,
        ', -112,0,1,0,1,0 ') ;
    END ( validationcell ) ;

PROCEDURE Read_data (VAR N_cell : N_Type) ;
VAR
  i,
  j,
  x      : Integer ;
  Ch     : Char ;
  Nor_in,
  a,
  b,
  c      : Str5 ;
  one,
  two,
  three  : Str4 ;

BEGIN
  For i := 1 to 16 do
    Begin
      For j := 1 to 4 do
        Begin
          Read (Nor_in, Ch) ;
          x := (i-1) * 4 + j ;
          Case j of
            4 :
              Begin
                a := 'NA.B ' ;

```

```

        b := 'NA' ;
        c := 'B' ;
        one := 'n10' ;
        two := 'n11' ;
        three := 'n12' ;
    End ;
3 :
    Begin
        a := 'NA.NB' ;
        b := 'NA' ;
        c := 'NB' ;
        one := 'n7' ;
        two := 'n8' ;
        three := 'n9' ;
    End ;
2 :
    Begin
        a := 'A.B' ;
        b := 'A' ;
        c := 'B' ;
        one := 'n4' ;
        two := 'n5' ;
        three := 'n6' ;
    End ;
1 :
    Begin
        a := 'A.NB' ;
        b := 'A' ;
        c := 'NB' ;
        one := 'n1' ;
        two := 'n2' ;
        three := 'n3' ;
    End ;
End ;
Validationcell (N_cell, Nor_in, a, b, c,
one, two, three, x) ;
End ; ( for j )
Readln ;
End ; ( for i )
End ; ( Read_data )

PROCEDURE Write_line ;

BEGIN
    Writeln ( '-----' ) ;
END ;

PROCEDURE Initialize_and ;

VAR
    i,
    j      : Integer ;
BEGIN ( ini )
    For i := 1 To AND_ROW DO

```

```

        For j := 1 To AND_COL DO
            with and_plane [ i,j ] DO
                cell := 'xxx';
END ; ( ini )
PROCEDURE Initialize_or ;
(
    In this procedure the OR_plane is initialized to xxx.
)
VAR
    i,
    j      : Integer ;
BEGIN ( ini )
    For i := 1 To OR_ROW DO
        For j := 1 To OR_COL DO
            with or_plane [ i,j ] DO
                cell := ' xxx ' ;
END ; ( ini )
PROCEDURE Print_and ;
(
    This procedure print
)
VAR
    i,
    j      : Integer ;
BEGIN ( print )
    write ( ' ' : 6)
    For j := AND_COL Downto 1 DO
        write ( ' ' : 4, j : 2);
    writeln ;
    For i := AND_ROW Downto 1 DO
        Begin
            write ( ' ' : 4, i : 3, ' ' : 2);
            For j := AND_COL Downto 1 DO
                write (and_plane[i,j].cell : 4,
                    ' ' : 2) ;
            writeln ;
        End ;
    END ;( print )
PROCEDURE Print_or ;
VAR
    i,
    j      : Integer ;
BEGIN ( print )
    write ( ' ' : 6);
    For j := 1 To OR_COL DO
        write ( ' ' : 4, j : 2) ;
    writeln ;
    For i := OR_ROW Downto 1 DO
        Begin
            write ( ' ' : 4, i : 3, ' ' : 2) ;
            For j := 1 To OR_COL DO
                write (or_plane[i,j].cell : 4,
                    ' ' : 2) ;
            writeln ;
        End ;
    End ;

```

```

END ; ( print )
PROCEDURE print_and_coordinates;
VAR
  i,
  j          : Integer ;
  row,
  column     : Real ;
BEGIN
  For i := 1 To AND_ROW DO
    Begin
      For j := 1 To AND_COL DO
        Begin
          with and_plane[ i,j ] DO
            Begin
              If cell <> 'xxx' Then
                Begin
                  writeln ('c ', cell : 2,
                    ', ', ( -625 + (( j-1)
                    Div 2) * 19) : 0, ', ', (
                    (250 - ((i-1) Div 2)
                    * 19)) : 0, ', 0,1,0,1,0');
                End ;
              End ;
            End ;
          End ;
        End ;
      End ;
    End ;
  End ;( print_and_coordinates )
PROCEDURE print_or_coordinates ;
VAR
  i,
  j          : Integer ;
  row1,
  column1    : Real ;
BEGIN
  For i := 1 To OR_ROW DO
    Begin
      For j := 1 To OR_COL DO
        Begin
          with or_plane[ i,j ] DO
            Begin
              If cell <> 'xxx' Then
                writeln ('c ', cell : 2,
                  ', ', ( 35 + (( j -
                  1) Div 2) * 22) : 0, ', ', (
                  ( -3 + ((i - 1) Div 2)
                  * 19) : 0, ', 0,1,0,1,0');
                End ;
              End ;
            End ;
          End ;
        End ;
      End ;
    End ;
  End ;(print_or_coordinates)
PROCEDURE And_Logic (VAR
  i,
  j : Integer ;
  a_input,

```

```

                                b_input : Char ) ;
BEGIN
  Case (j Mod 4) of
    1 :
      If ((a_input = '0')
        And
        (b_input = '1')
        And
        (n_cell[x] = 'n1 '))
      Or
      ((a_input = '0')
        And
        (n_cell[x] = 'n2 '))
      Or
      ((b_input = '1')
        And
        (n_cell[x] = 'n3 ')) Then
        Begin ( 'ab' )
          With and_plane[ i,j ] Do
            Begin
              If (i Mod 2) = 1 Then
                cell := 'ac4 '
              Else
                cell := 'ac2 ' ;
              row := i;
              column := j;
            End ;
          End ;( 'ab' )
        End ;
    2 :
      If ((a_input = '0')
        And
        (b_input = '0')
        And
        (n_cell[x] = 'n4 '))
      Or
      ((a_input = '0')
        And
        (n_cell[x] = 'n5 '))
      Or
      ((b_input = '0')
        And
        (n_cell[x] = 'n6 ')) Then
        Begin ( 'a'b' )
          With and_plane[ i,j ] Do
            Begin
              If (i Mod 2) = 1 Then
                cell := 'ac3 '
              Else
                cell := 'ac1 ' ;
              row := i;
              column := j;
            End ;
          End ;( 'a' b' )
        End ;
    3 :

```

```

If ((a_input = '1')
And
(b_input = '1')
And
(n_cell[x] = 'n7'))
Or
((a_input = '1')
And
(n_cell[x] = 'n8'))
Or
((b_input = '1')
And
(n_cell[x] = 'n9')) Then
Begin ( ab )
With and_plane [ i,j ] Do
Begin
If (i Mod 2) = 1 Then
cell := 'ac4 '
Else
cell := 'ac2 ' ;
row := i;
column := j;
End ;
End ;( ab )
0 :
If ((a_input = '1')
And
(b_input = '0')
And
(n_cell[x] = 'n10'))
Or
((a_input = '1')
And
(n_cell[x] = 'n11'))
Or
((b_input = '0')
And
(n_cell[x] = 'n12')) Then
Begin ( a'b )
With And_plane[ i,j ] Do
Begin
If (i Mod 2) = 1 Then
cell := 'ac3 '
Else
cell := 'ac1 ' ;
row := i;
column := j;
End ;
End ;(a' b)
End ;(* case *)
END ;( Logic )

PROCEDURE Or_Logic (VAR
i,

```

```

                                j,
                                aa_output : Integer );
BEGIN
  Case (i Mod 2) Of
    1 :
      If aa_output = 1 Then
        With or_plane[ i,j ] do
          Begin
            If (j Mod 2) = 0 Then
              cell := 'oc4'
            Else
              cell := 'oc3';
            End ;
          End ;
        End ;
      End ;
    0 :
      If aa_output = 1 Then
        With or_plane[ i,j ] do
          Begin
            If (j Mod 2) = 0 Then
              cell := 'oc2'
            Else
              cell := 'oc1';
            End ;
          End ;
        End ;
      End ;
  End ;(case)
END ;

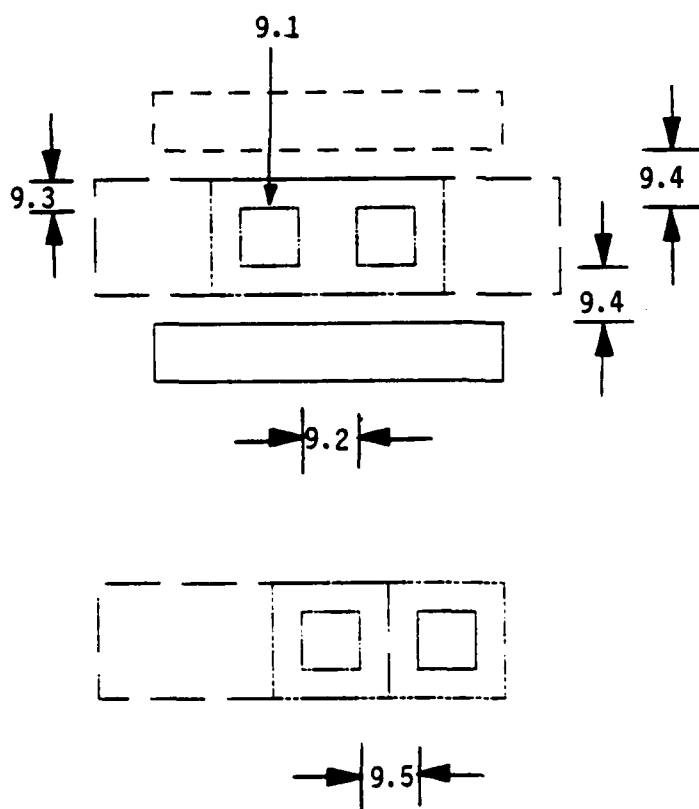
BEGIN (* main *)
  Read_data (n_cell);
  Write_line ;
  Initialize_and;
  Write_line ;
  For i := 1 To AND_ROW do
    Begin ( i )
      For j := 1 To AND_COL do
        Begin ( j )
          x := j;
          If ((j-1) Mod 4) = 0 then
            Begin
              Read (a_input, ch, b_input, ch);
            End ;
            And_logic (i, j, a_input, b_input);
          End ;( j )
        End ;
      End ;
    End ( i )
  End ;
  Writeln ;
  Writeln ;
  Initialize_or ;
  Write_line ;
  Writeln ('call cell for AND_array ');
  Write_line ;
  Print_And_coordinates;
  Write_line ;
  Writeln ;
  For i := 1 To OR_ROW do
    Begin

```



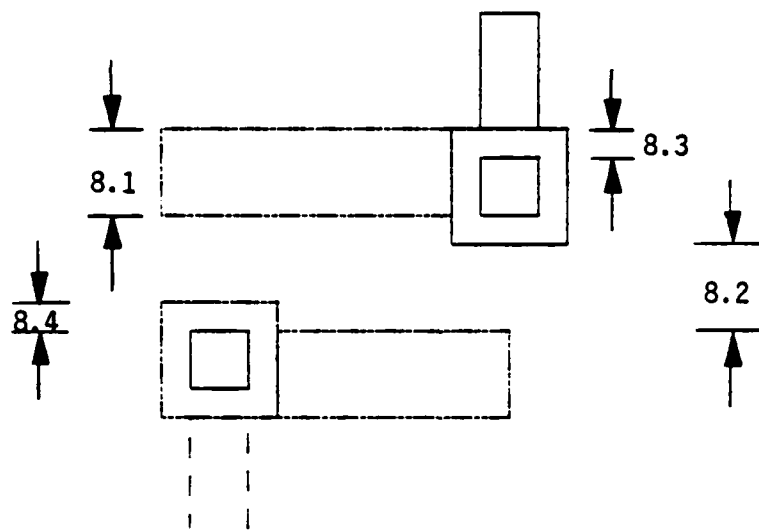
B.9 VIA

	LAMBDA
9.1 VIA SIZE . . . . .	2X2
9.2 VIA SEPERATION . . . . .	2
9.3 METAL 2 OVERLAP. . . . .	1
9.4 SPACE TO POLY OR ACTIVE EDGE . . . . .	2
9.5 SPACE TO CONTACT . . . . .	2



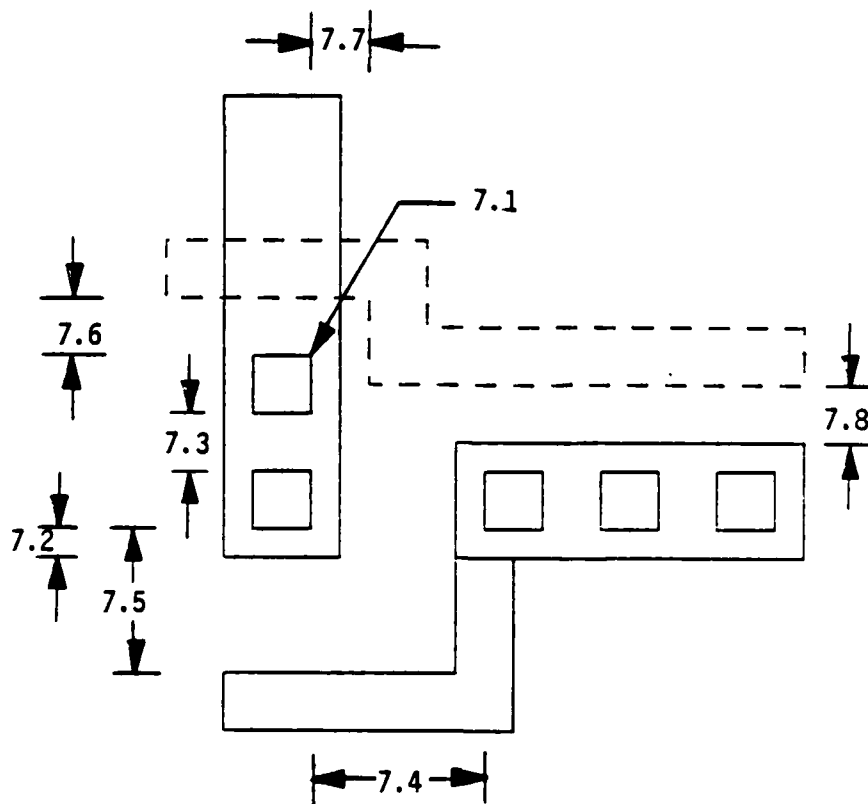
B.8 METAL 1

	LAMBDA
8.1 MINIMUM WIDTH . . . . .	3
8.2 MINIMUM SPACE . . . . .	3
8.3 MINIMUM POLY CONTACT OVERLAP. . . . .	1
8.4 MINIMUM ACTIVE CONTACT OVERLAP. . . . .	1



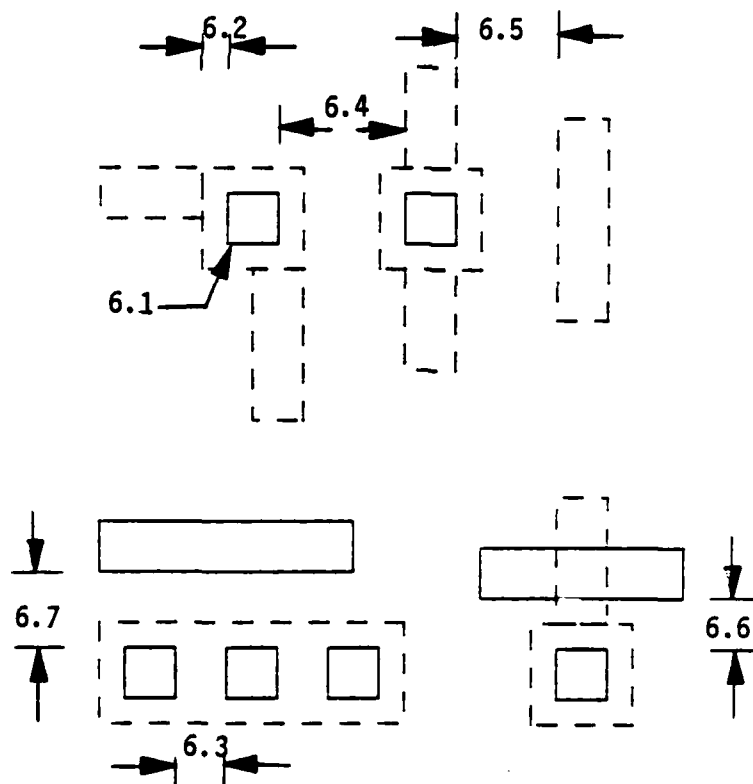
7.1 CONTACT TO ACTIVE

	LAMBDA
7.1 CONTACT SIZE . . . . .	2X2
7.2 ACTIVE OVERLAP . . . . .	1
7.3 SPACING ON SAME ACTIVE . . . . .	2
7.4 SPACING ON DIFFERENT ACTIVE. . . . .	6
7.5 SPACING TO DIFFERENT ACTIVE. . . . .	5
7.6 SPACE TO GATE. . . . .	2
7.7 SPACE TO POLY OVER FIELD, SHORT RUN. . . . .	2
7.8 SPACE TO POLY OVER FIELD, LONG RUN . . . . .	3



B.6 CONTACT TO POLY

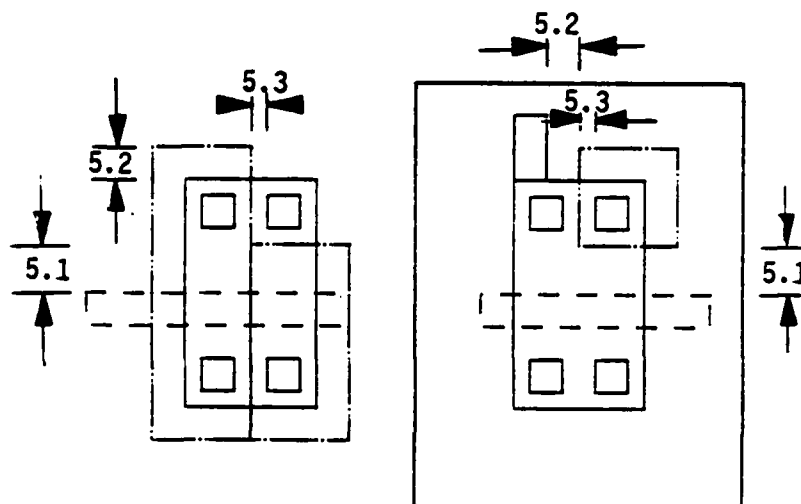
	LAMBDA
6.1 CONTACT SIZE . . . . .	2X2
6.2 POLY OVERLAP OF CONTACT. . . . .	1
6.3 SPACING ON SAME POLY . . . . .	2
6.4 SPACING ON DIFFERENT POLY. . . . .	5
6.5 SPACE TO OTHER POLY. . . . .	4
6.6 SPACE TO ACTIVE, SHORT RUN . . . . .	2
6.7 SPACE TO ACTIVE, LONG RUN . . . . .	3



B.5 PSELECT

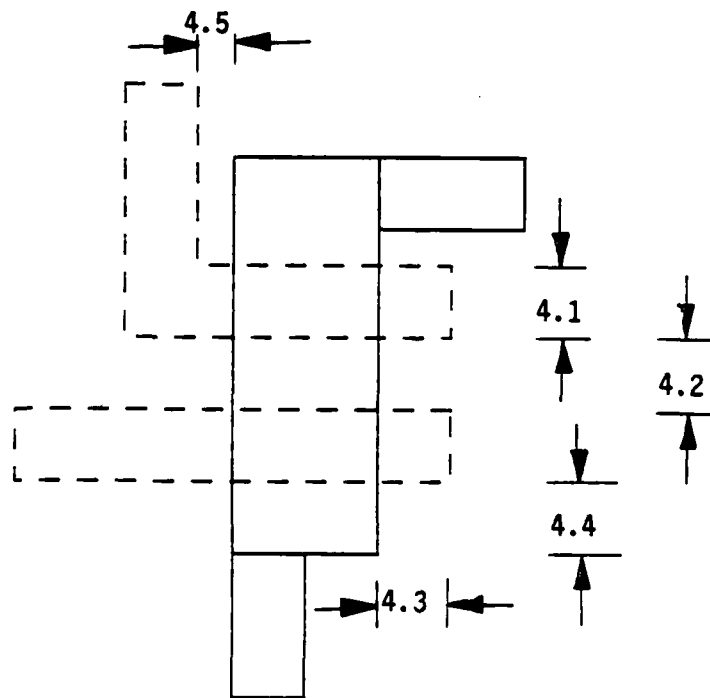
LAMBDA

5.1	PSELECT SPACE (OVERLAP) TO (OF) GATE. . . .	3
5.2	PSELECT SPACE (OVERLAP) TO (OF) ACTIVE. . .	2
5.3	PSELECT SPACE (OVERLAP) TO (OF) CONTACT TO SUBSTRATE . . . . .	1



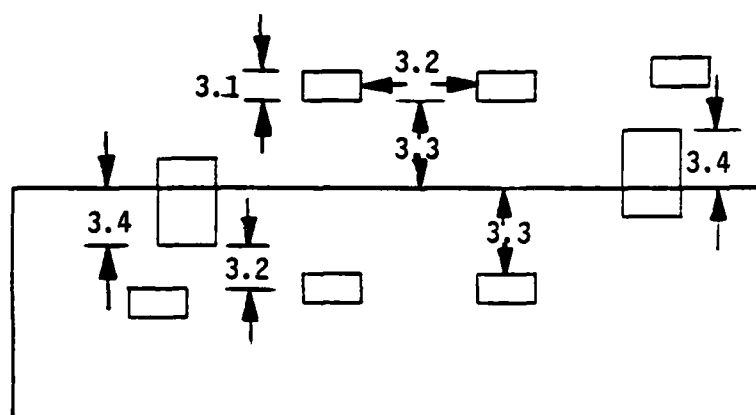
B.4 POLYSILICON

	LAMBDA
4.1 MINIMUM WIDTH . . . . .	2
4.2 MINIMUM SPACE . . . . .	2
4.3 GATE OVERLAP OF ACTIVE. . . . .	2
4.4 ACTIVE OVERLAP OF GATE. . . . .	2
4.5 FIELD POLY TO ACTIVE. . . . .	1



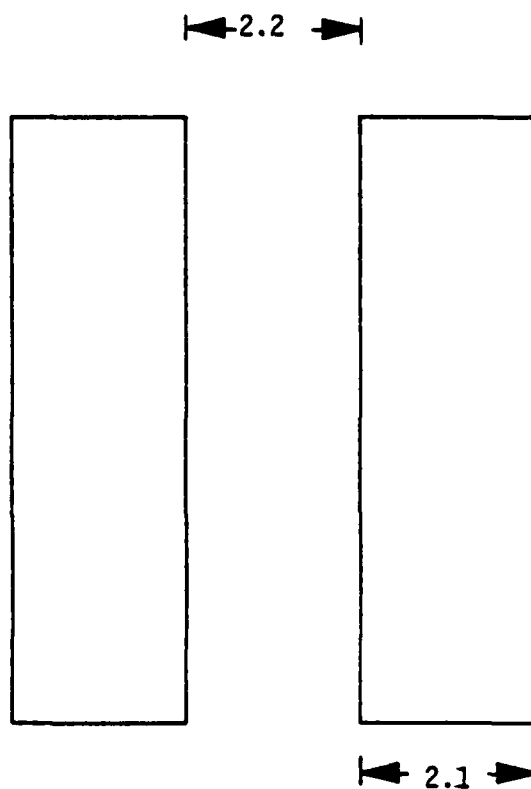
B.3 ACTIVE

	LAMBDA
3.1 MINIMUM WIDTH . . . . .	2
3.2 MINIMUM SPACE . . . . .	3
3.3 SOURCE/DRAW ACTIVE TO WELL EDGE . . . . .	6
3.4 SUBSTRATE CONTACT ACTIVE OVERLAP OF SUBSTRATE . . . . .	4

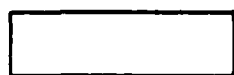


B.2 P-WELL (N-WELL)

	LAMBDA
2.1 WELL WIDTH . . . . .	6
2.2 WELL SPACE . . . . .	6



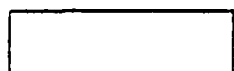


B.1 LEGEND

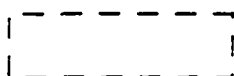
P well



P select



Active



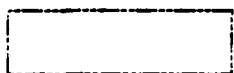
Polysilicon



Contact to poly



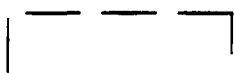
Contact to active



First metal



Via



Second metal

The more debatable rule is the use of stacked vias. The contact from second metal to active with a stacked via is illustrated in B-11. Also shown are other versions of designs which abide by the rules. Since control lines are most often implemented in the first metal in order to connect to poly gates conveniently, the second metal often must connect to active for data lines and supply lines. A possible compromise on the rules would be to require only a one lambda active overlap of the via, taking into account the less steep edge of the active due to the commonly used selective oxidation processing. This would not require analyzing the active pad. With this compromise, the PPL core cell is increased approximately 10% over one with the stacked via.

Thus the stacked via is most convenient and results in smaller layouts. MOSIS needs to support the stacked via.

### DESIGN RULE ISSUES

The lambda based scalable design rules presented here have been used for this project. Lambda is equal to 1.5 microns for 3 micron technology, and 0.8 microns for 1.2 micron technology. A few exceptions have been made, however, because the project started well before the rules came into effect. The exceptions are:

- 1) The spacing between two different active regions is three lambda instead of four lambda.
- 2) The first metal minimum width is two lambda instead of three lambda.
- 3) Stacked vias (via over a contact) are used for the designs.

The first exception can be accommodated without much loss in the area, but the next two need consideration. Since the first metal is used for local interconnect, and in most designs limits the density of the layout, it is very desirable to have a two lambda minimum width. Silicon-gate CMOS without buried contacts places an additional interconnect burden on the first metal. Even using a two lambda first metal minimum width with the scalable rules, the first metal minimum pitch is 7.5 microns for the 3-micron process, compared with the MOSIS standard of 7 microns. So a first metal minimum width of two lambda is a reasonable compromise.

CONTENTS

	Page
Design rule issues . . . . .	117
B-1 Legend. . . . .	119
B-2 P-well. . . . .	120
B-3 Active. . . . .	121
B-4 Polysilicon . . . . .	122
B-5 P-select. . . . .	123
B-6 Contact to Poly . . . . .	124
B-7 Contact to Active . . . . .	125
B-8 First Metal . . . . .	126
B-9 Via . . . . .	127
B-10 Second Metal. . . . .	128
B-11 Glass . . . . .	129
B-12 Via contacts. . . . .	130

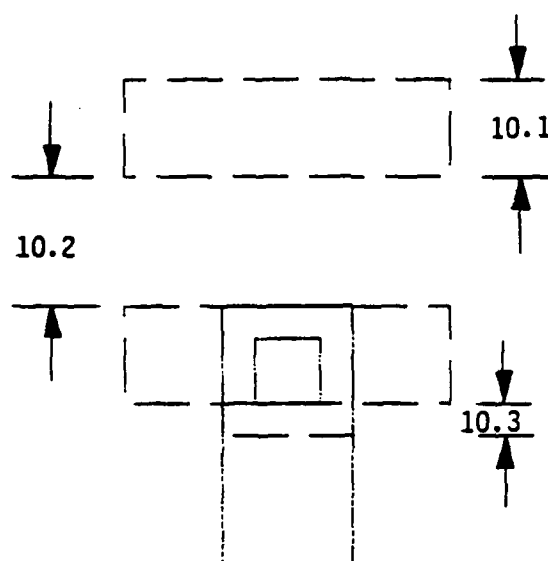
APPENDIX B

SCALABLE DESIGN RULES

```
For j := 1 To OR_COL do
Begin
    Read (aa_output);
    Or_logic (i, j, aa_output)
End ;
End ;
Write_line ;
WriteLn ('call cells for Or_array ');
Write_line ;
WriteLn ;
Print_or_coordinates;
WriteLn ;
Write_line ;
END .
```

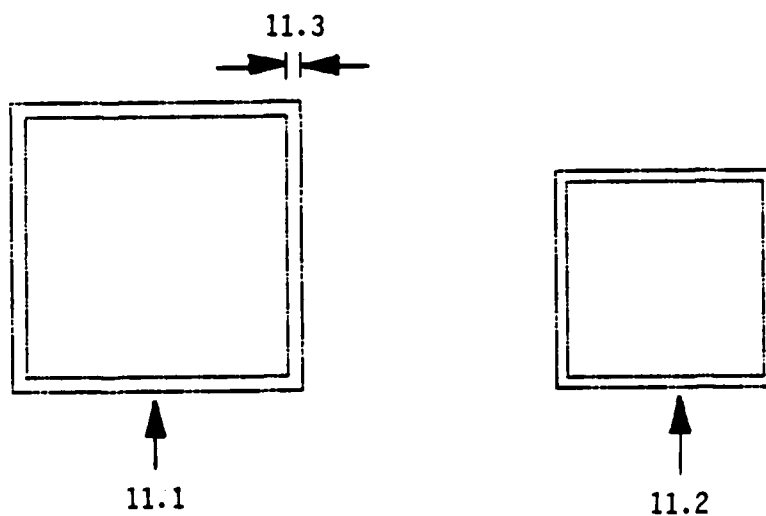
B.10 METAL 2

	LAMBDA
10.1 MINIMUM WIDTH . . . . .	3
10.2 MINIMUM SPACE . . . . .	4
10.3 MINIMUM VIA OVERLAP . . . . .	1

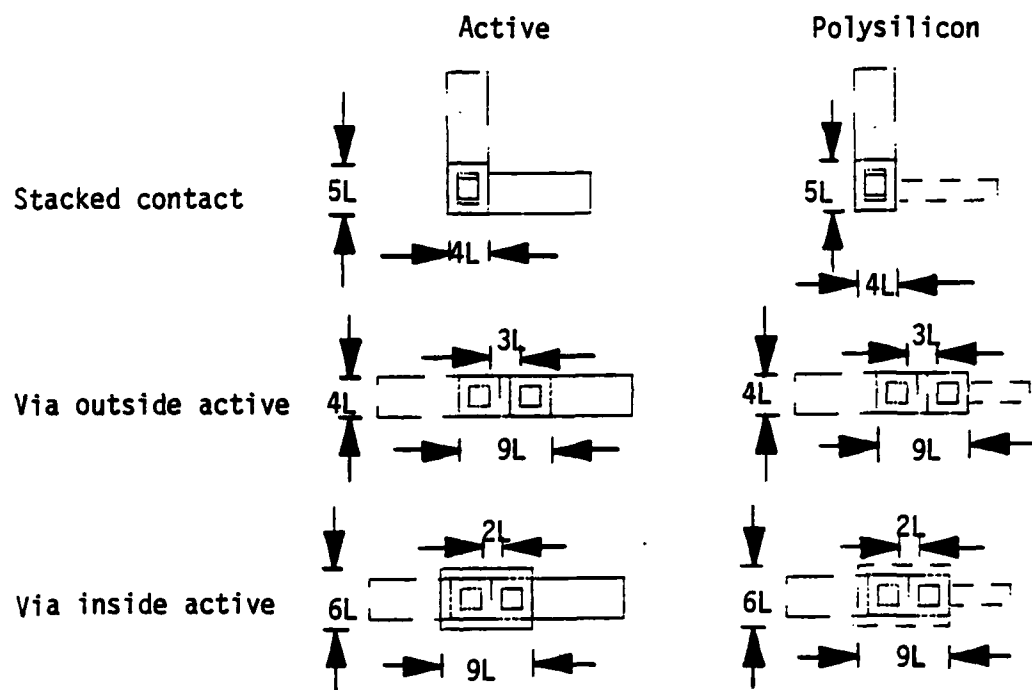


B.11 GLASS

	LAMBDA
11.1 MINIMUM BONDING PAD. . . . .	100 x 100
11.2 MINIMUM PROBE PAD. . . . .	75 x 75
11.3 PAD TO GLASS EDGE. . . . .	5





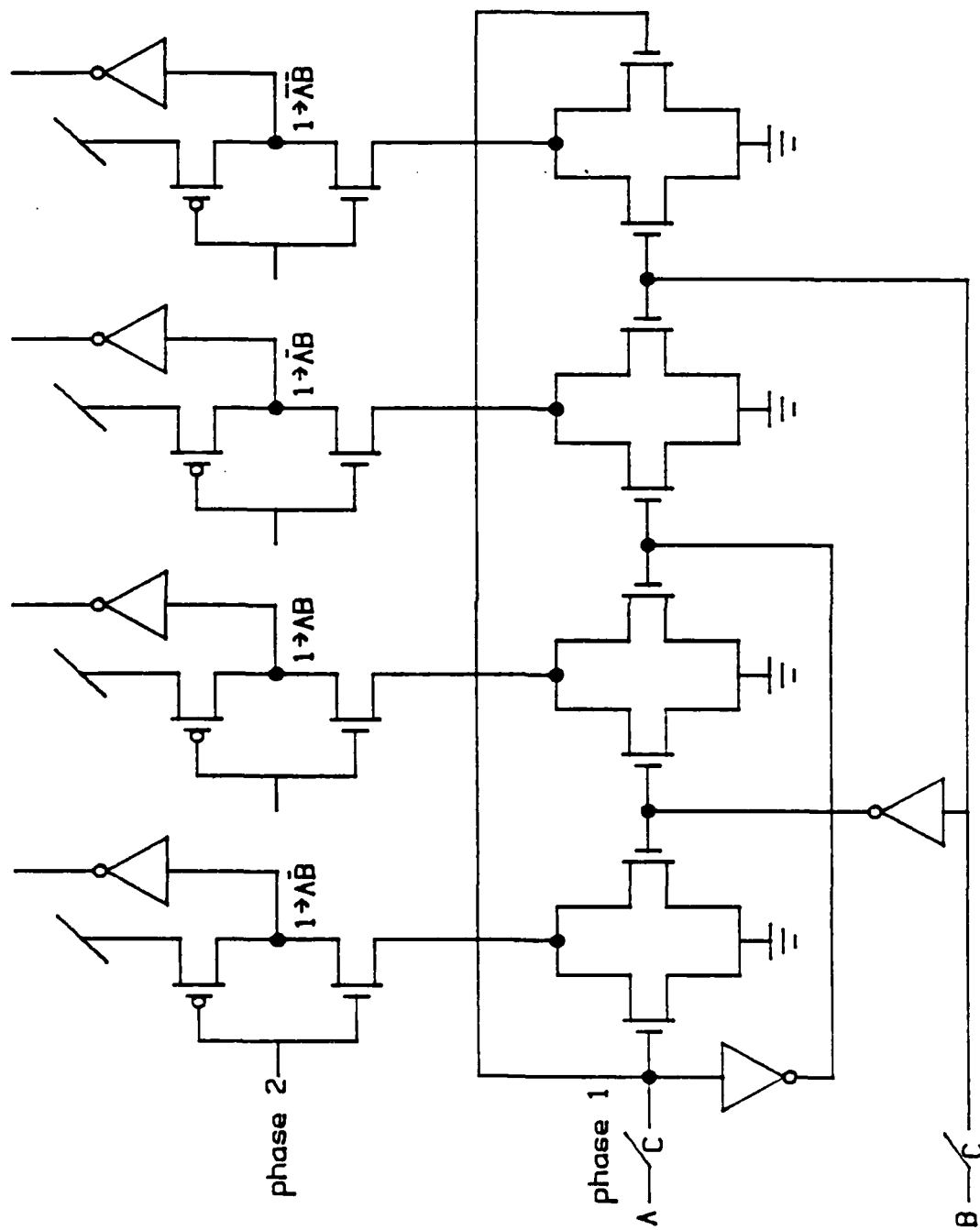
B.12 VIA CONTACTS

APPENDIX C

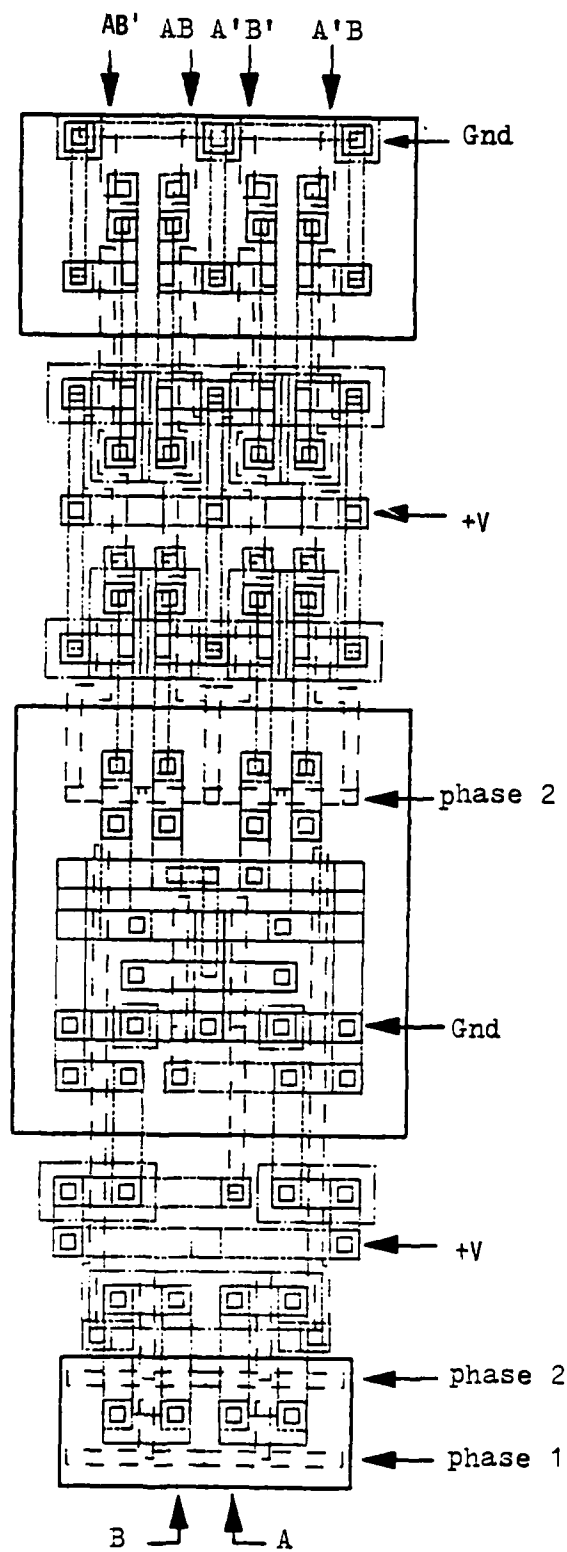
PLA'S CELL LIBRARY

CONTENTS

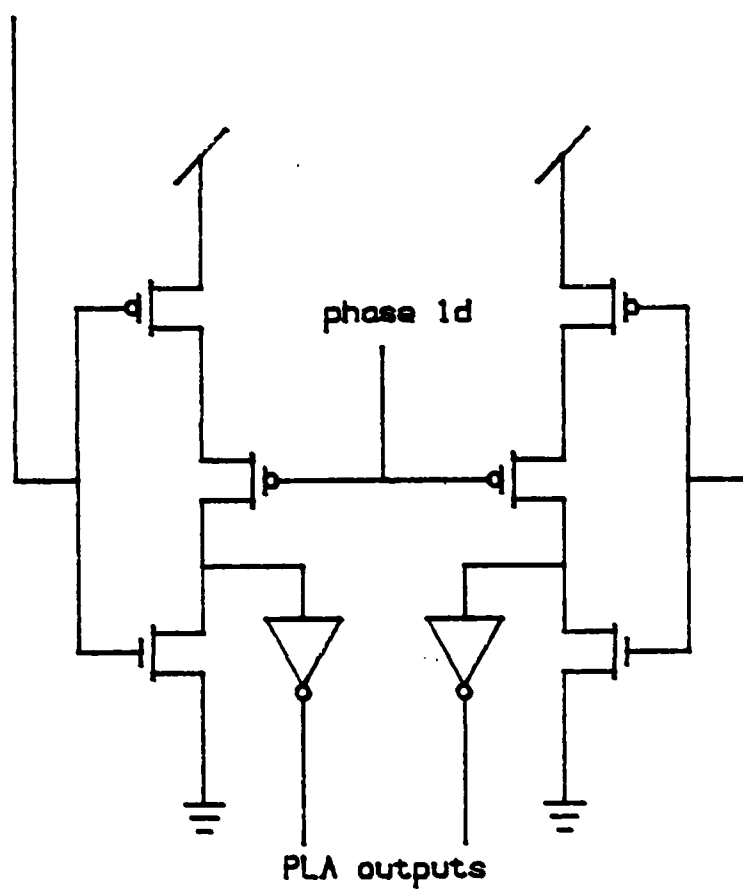
	Page
C.1 Input latch for large PLA. . . . .	133-134
C.2 Output latch for large PLA . . . . .	135-136
C.3 AND-OR for large PLA . . . . .	137-138
C.4 Set-Reset latch for large PLA. . . . .	139-140
C.5 Delay signal generator for large PLA . . .	141-142
C.6 Input latch for moderate PLA . . . . .	143-144
C.7 Output latch for moderate PLA. . . . .	145-146
C.8 AND plane circuit for moderate PLA . . . .	147-148
C.9 OR plane circuit for moderate PLA. . . . .	149-150
C.10 Input latch for small PLA. . . . .	151-152
C.11 Output latch for small PLA . . . . .	153-154
C.12 AND plane circuit for small PLA. . . . .	155-156
C.13 OR plane circuit for small PLA . . . . .	157-158



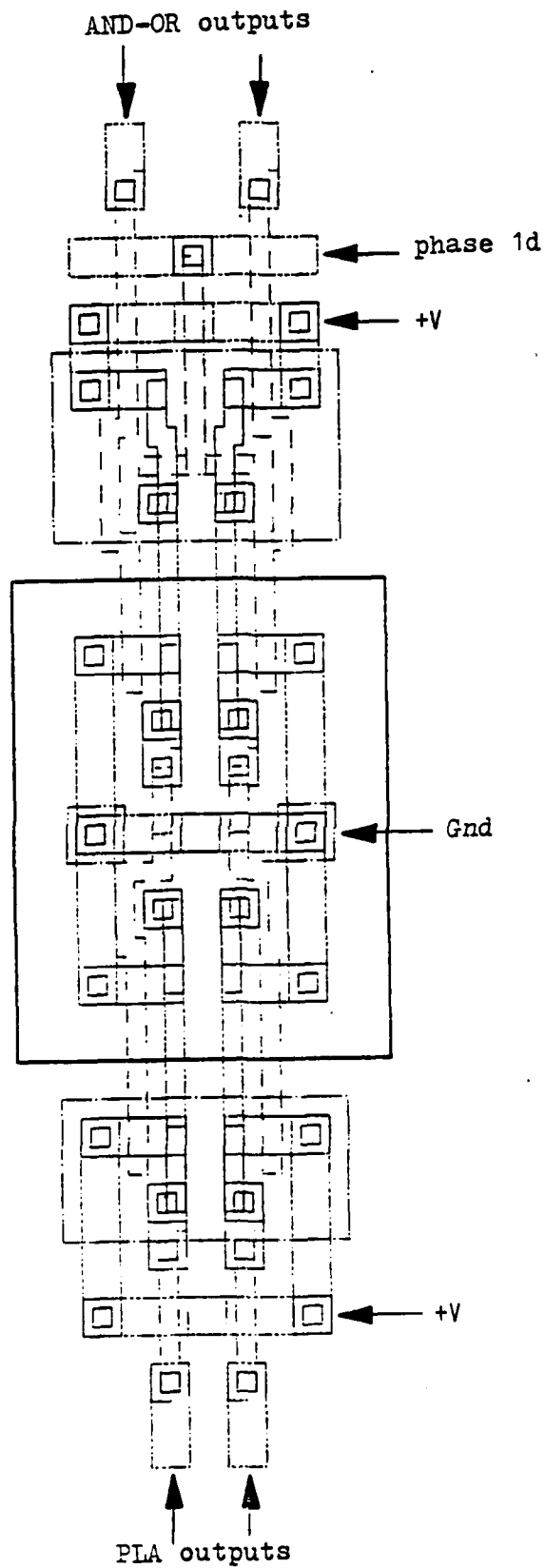
C.1 Input latch for large PLA



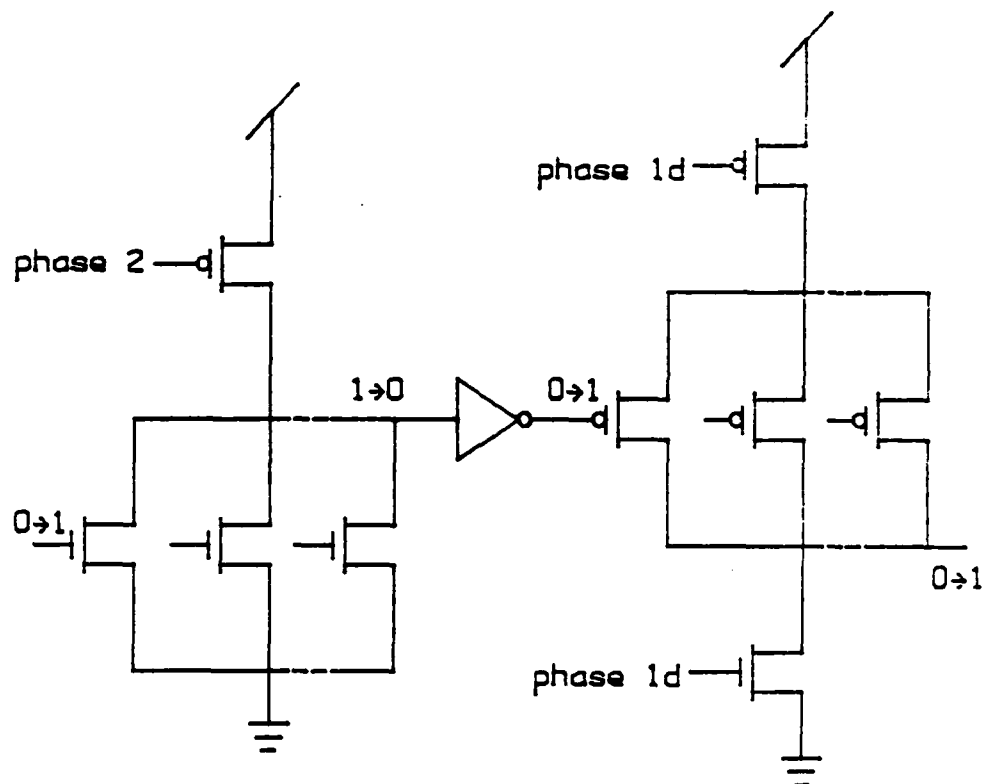
C.1 Input latch for large PLA



C.2 Output latch for large PLA

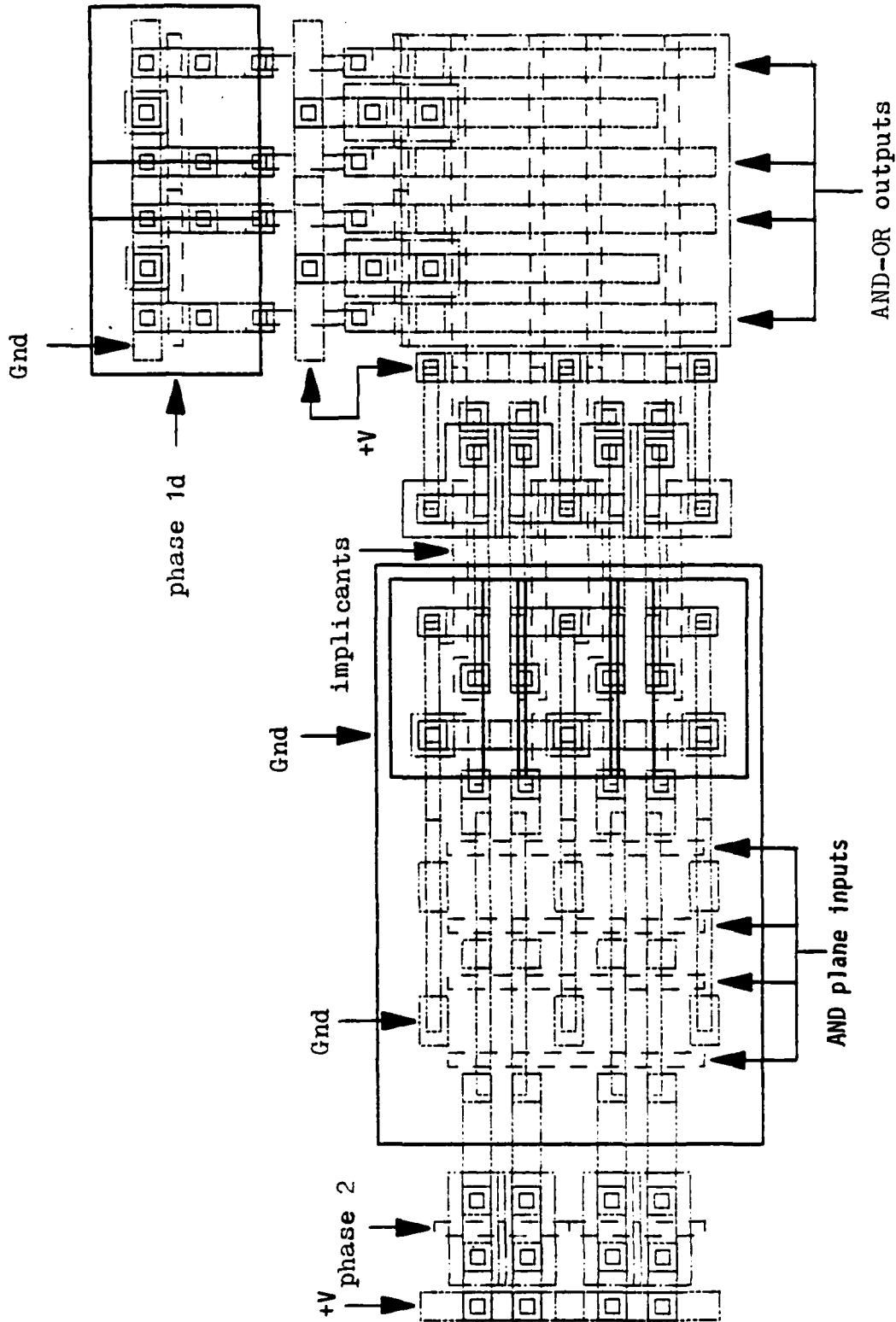


C.2 Output latch for large PLA

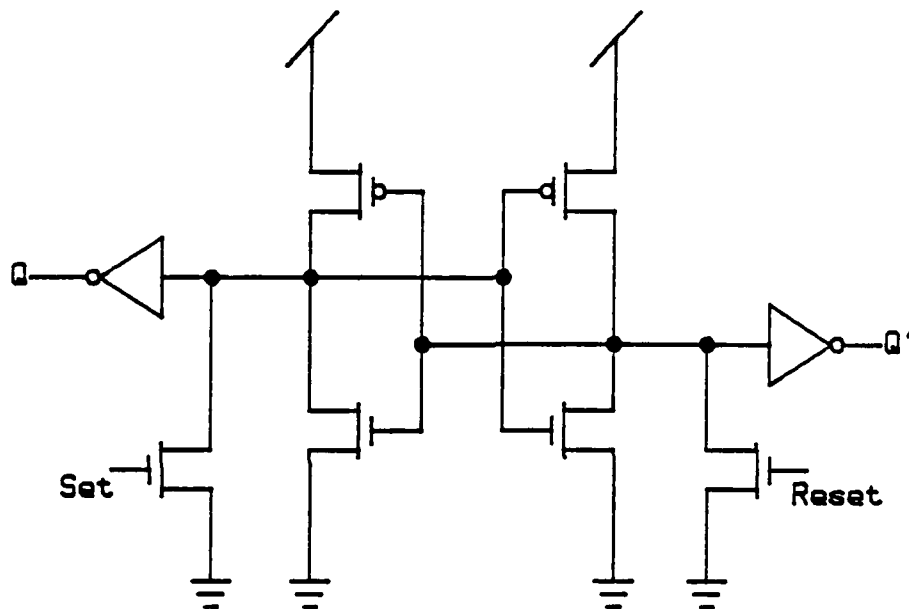


C.3 AND-OR for large PLA

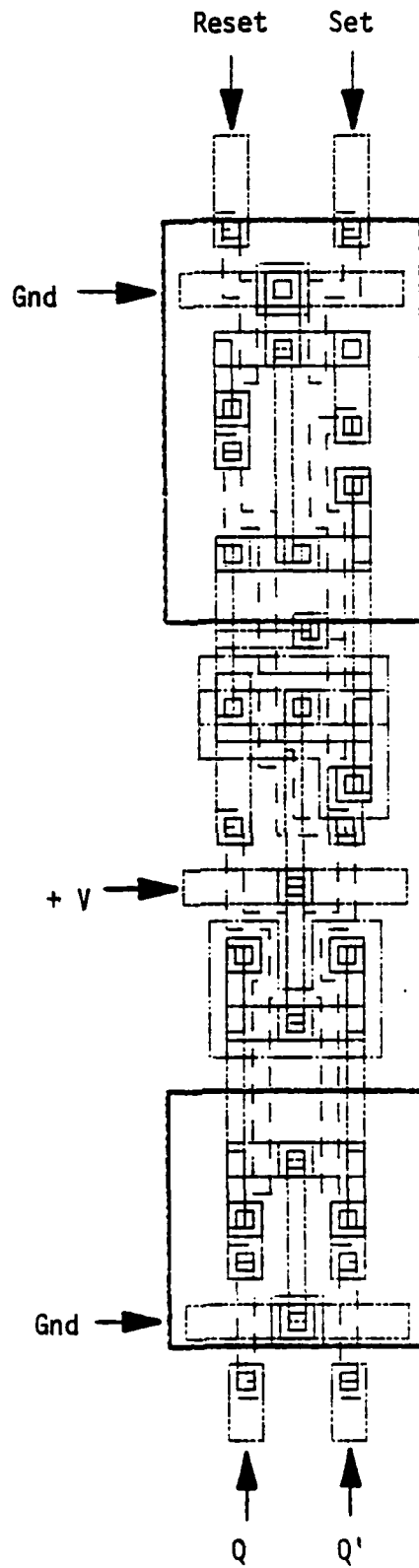




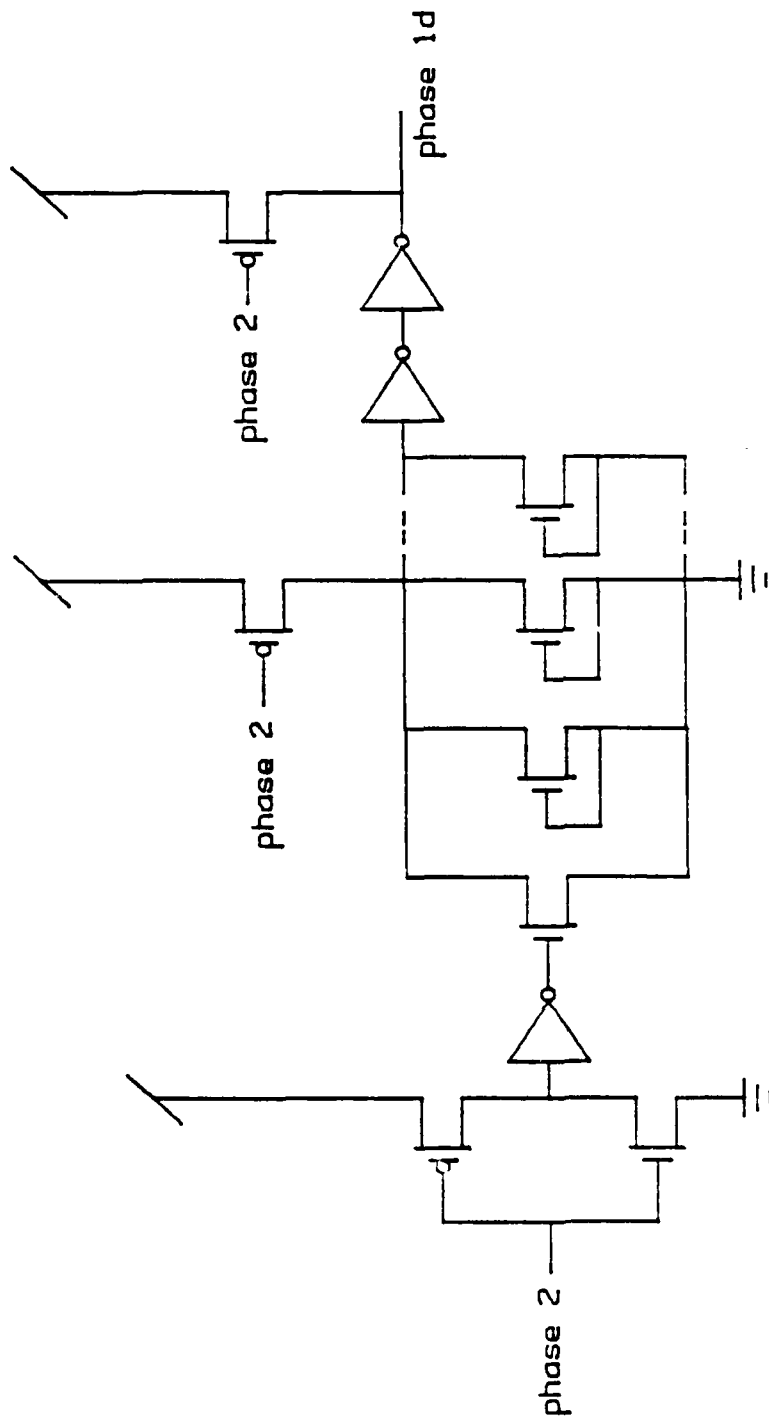
C.3 AND-OR for large PLA



C.4 Set-Reset latch for large PLA



C.4 Set-Reset latch for large PLA



C.5 Delay signal generator for large PLA

AD-A158 367

BULK CMOS VLSI TECHNOLOGY STUDIES PART 1 SCALABLE CMOS  
DESIGN RULES PART 2. (U) MISSISSIPPI STATE UNIV  
MISSISSIPPI STATE DEPT OF ELECTRICAL E.

3/3

**UNCLASSIFIED**

MISSISSIPPI STATE DEPT OF ELI  
J D TROTTER ET AL. 17 JUN 85

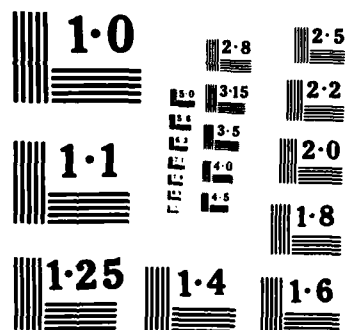
F/G 9/5

NL

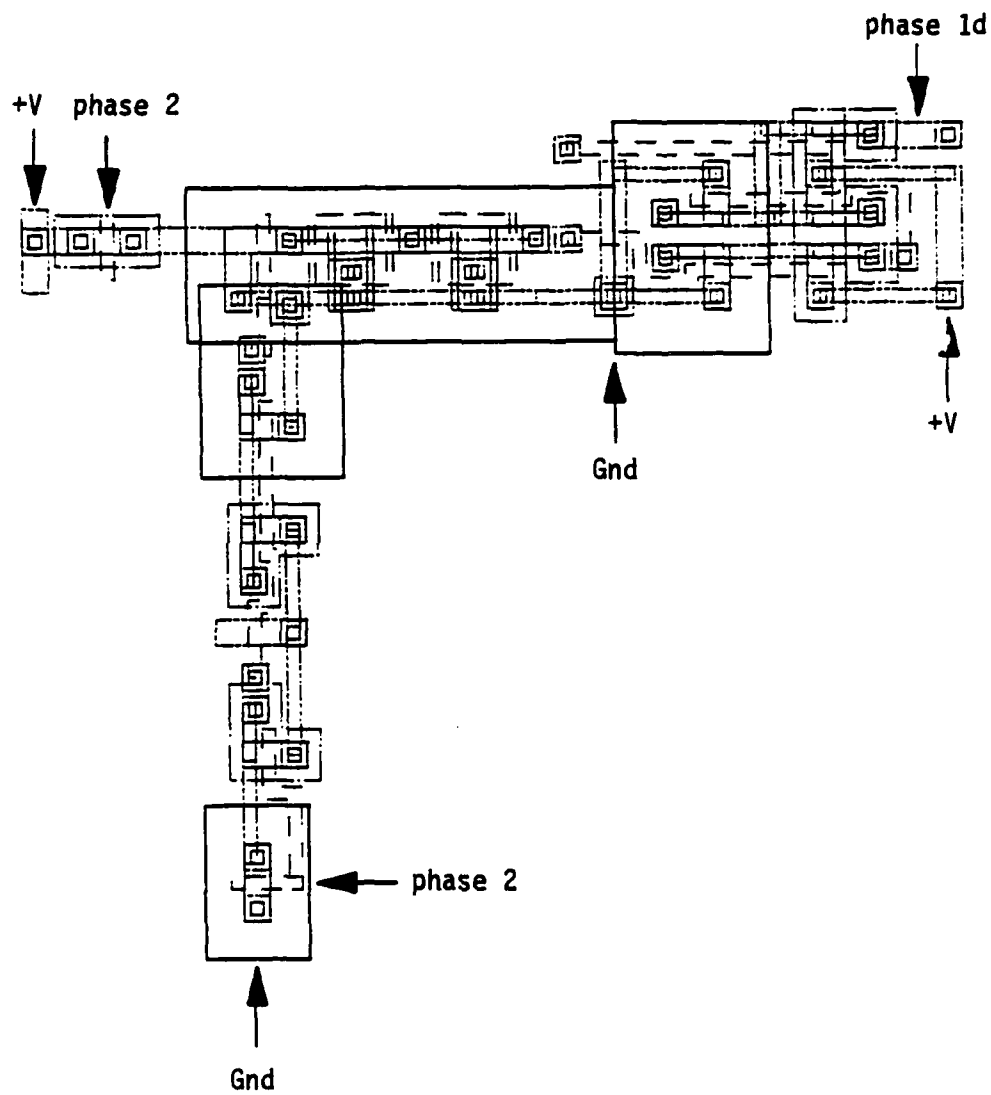
END

FILMED

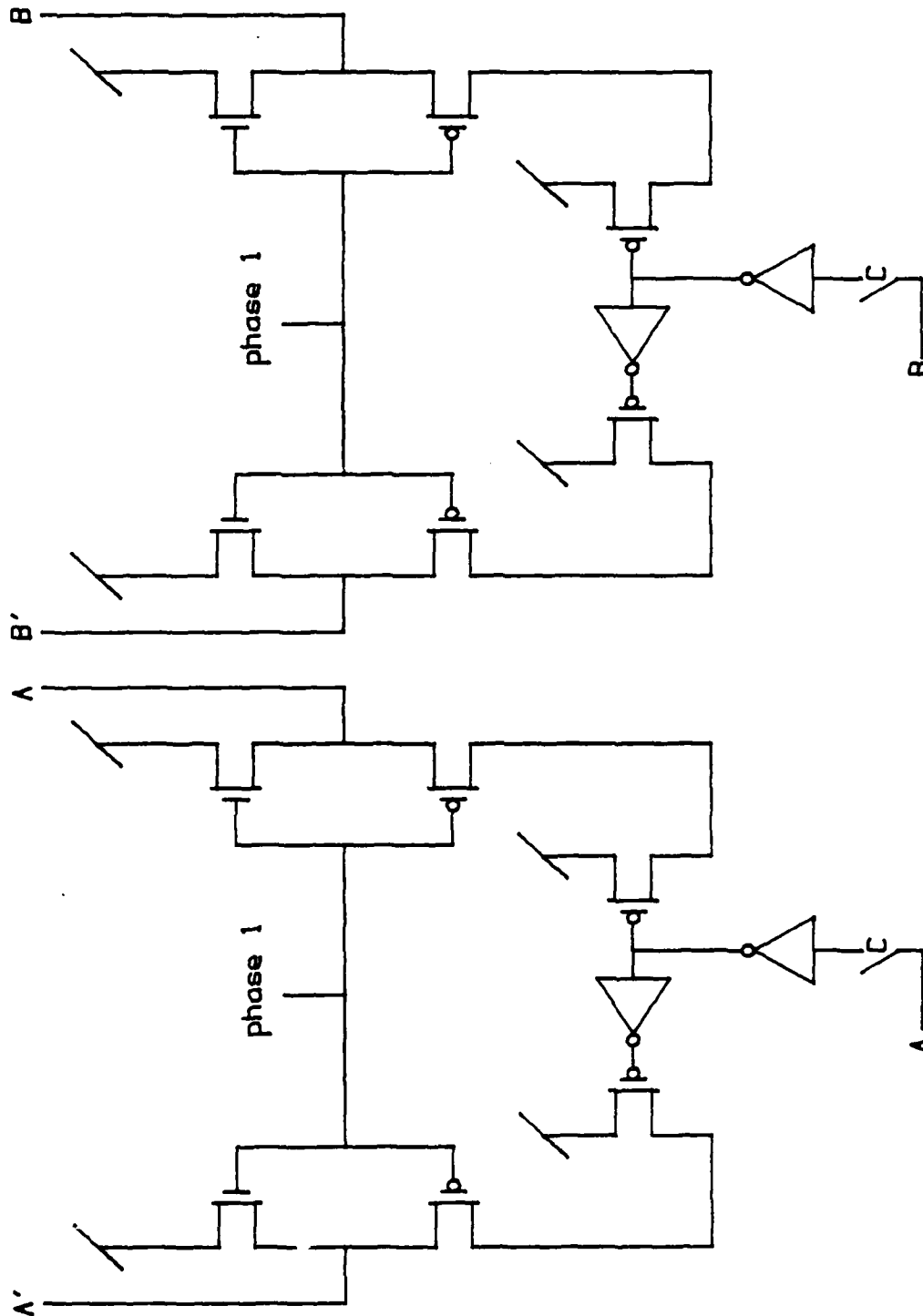
DTIC



NATIONAL BUREAU OF STANDARDS  
MICROCOPY RESOLUTION TEST CHART

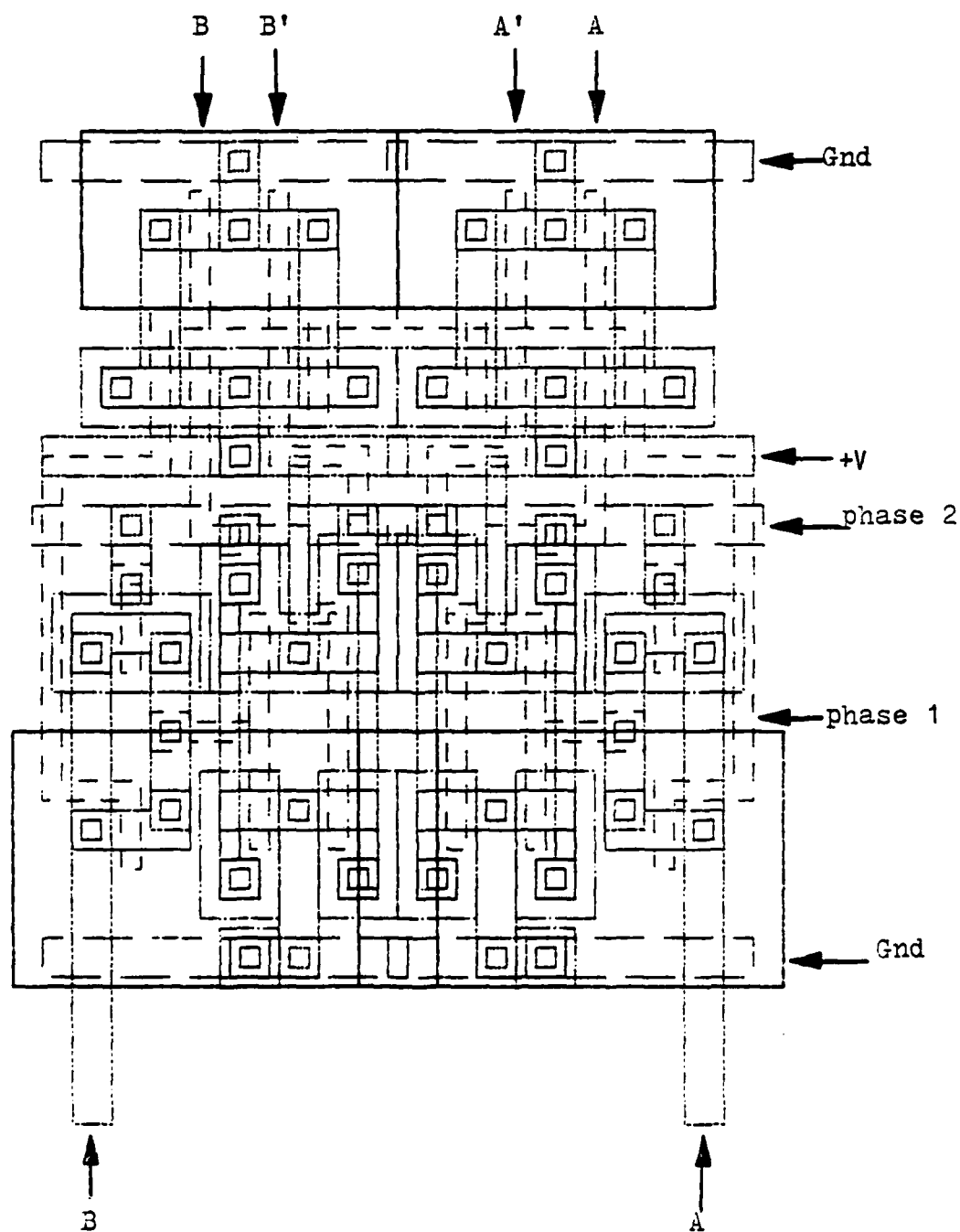


C.5 Delay signal generator for large PLA

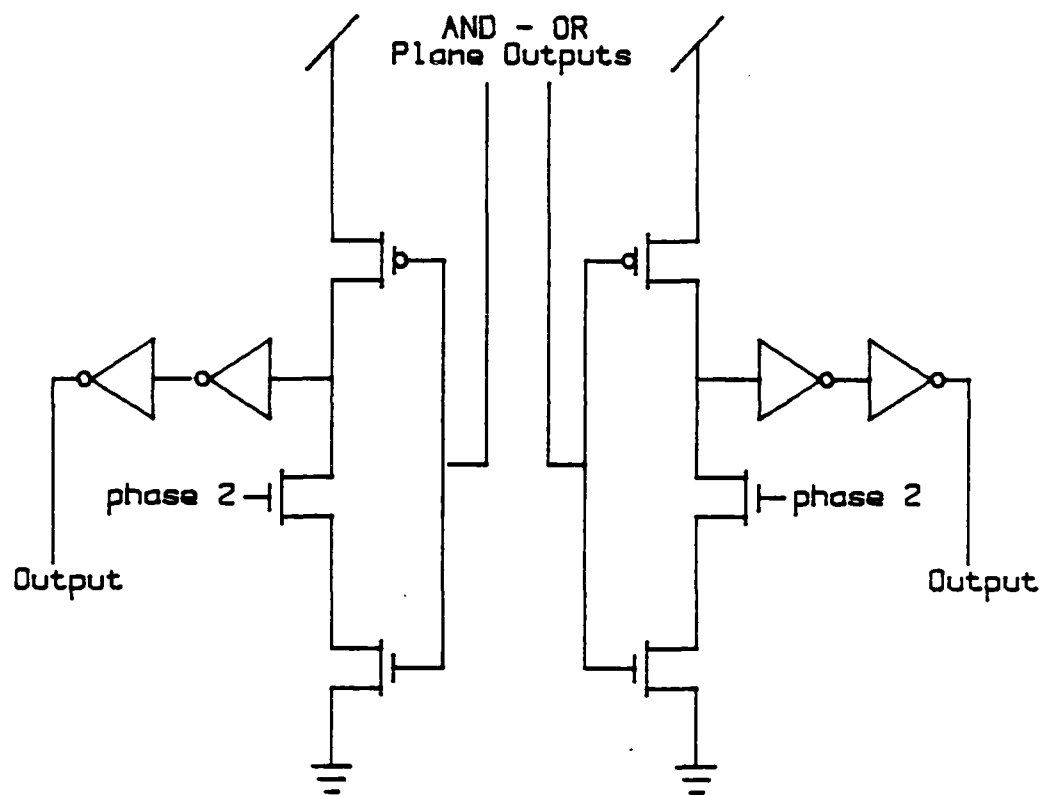


C.6 Input latch for moderate PLA

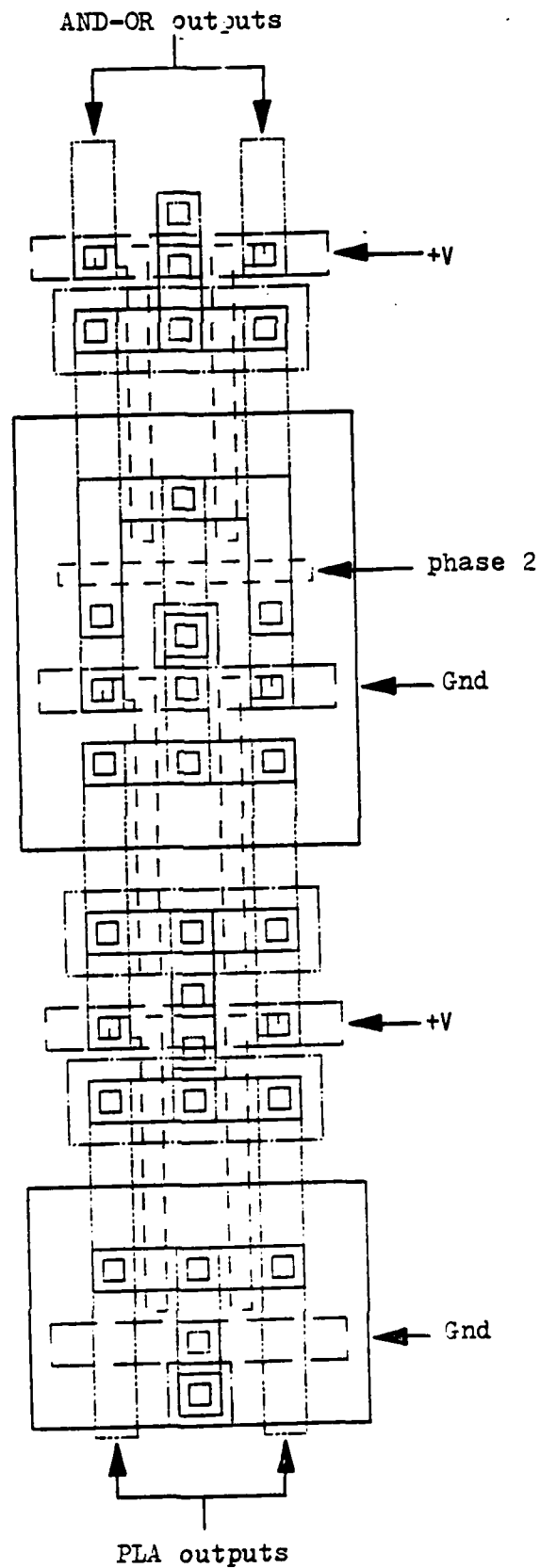




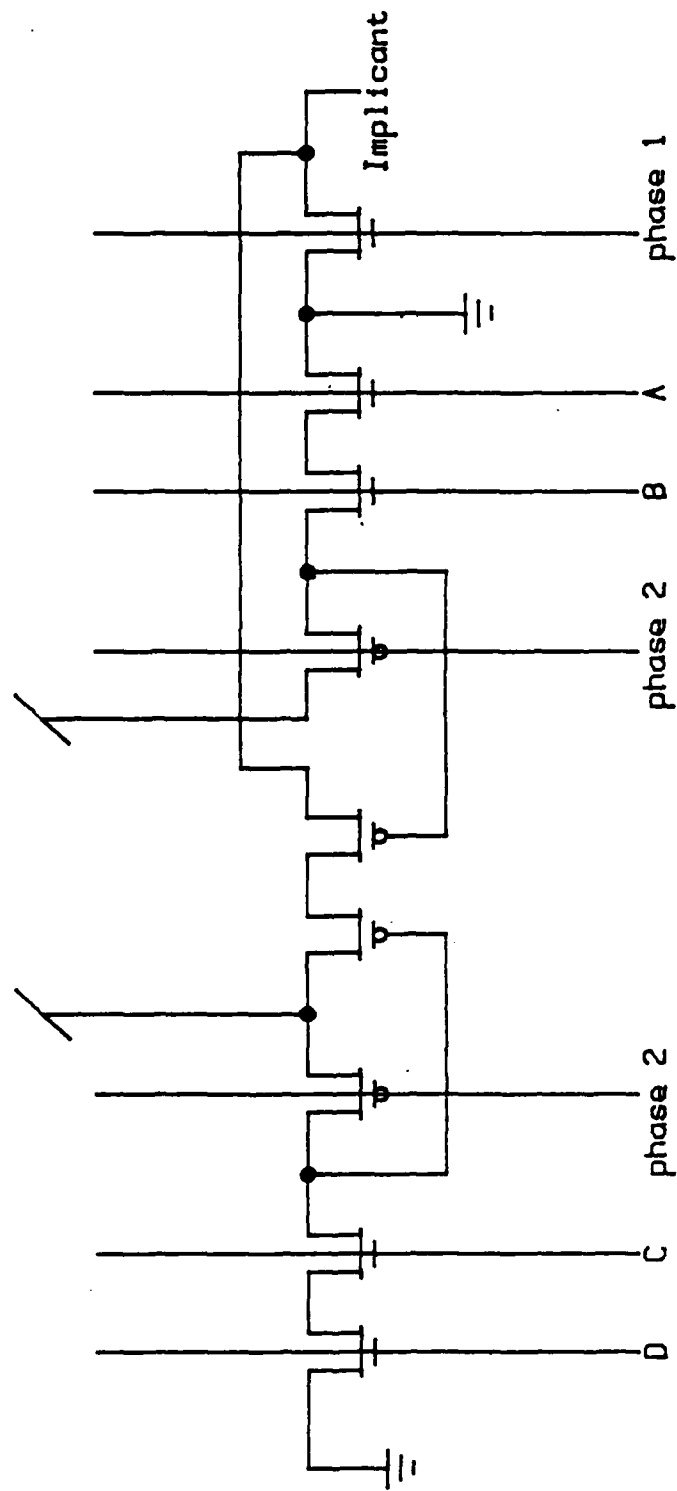
C.6 Input latch for moderate PLA



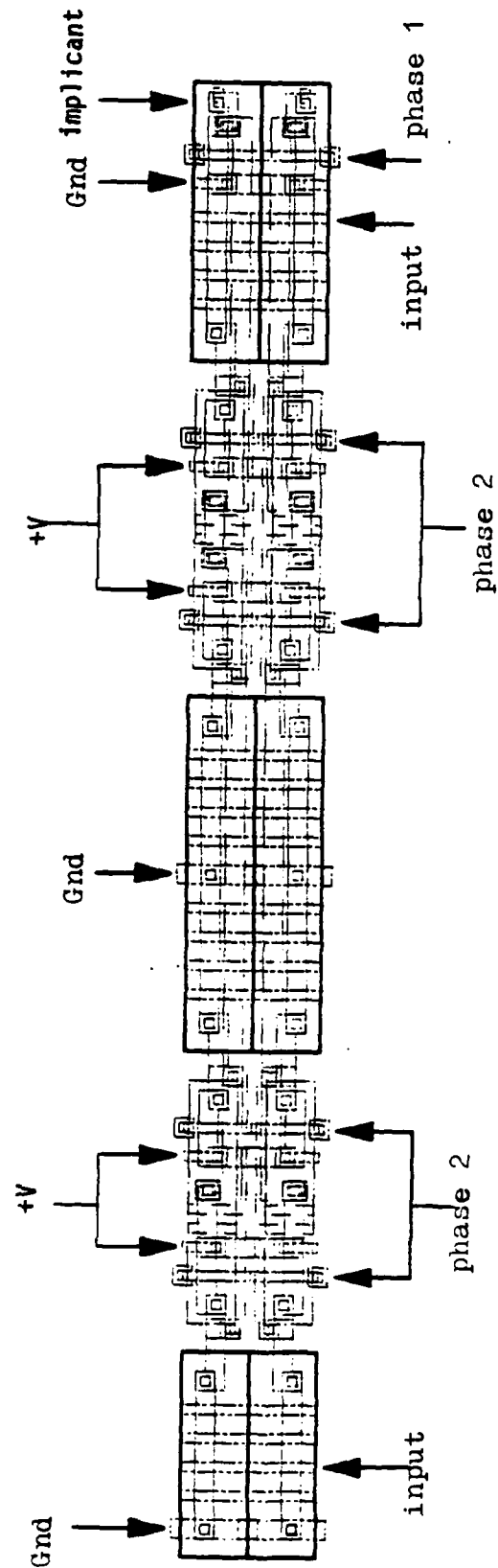
C.7 Output latch for moderate PLA



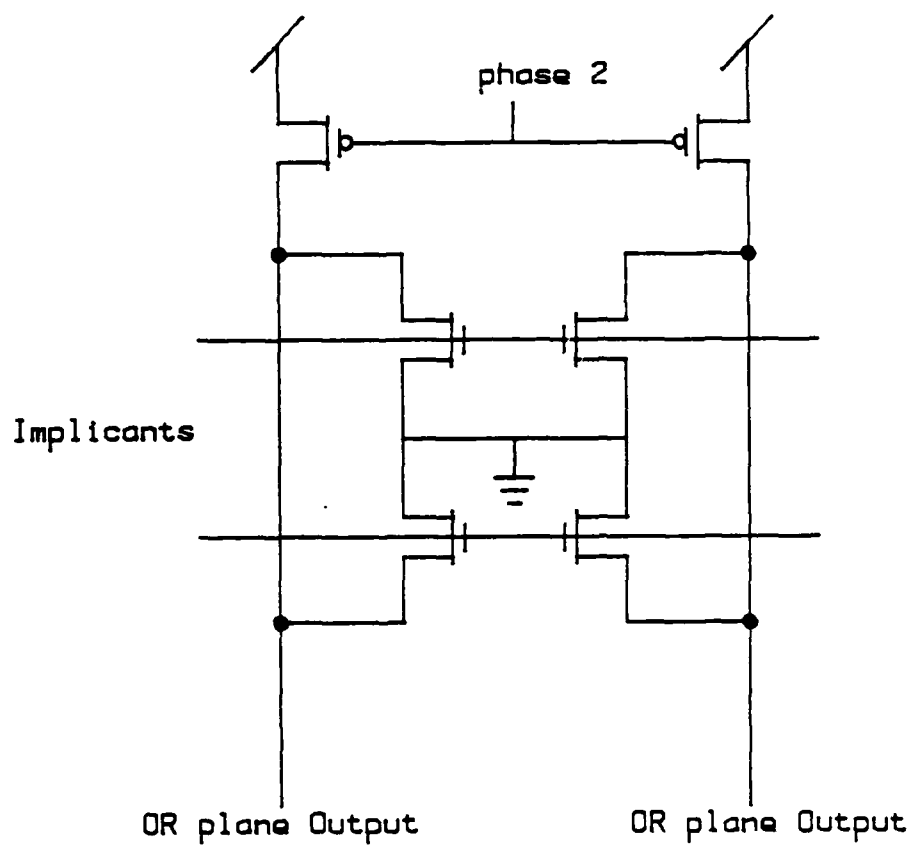
C.7 Output latch for moderate PLA



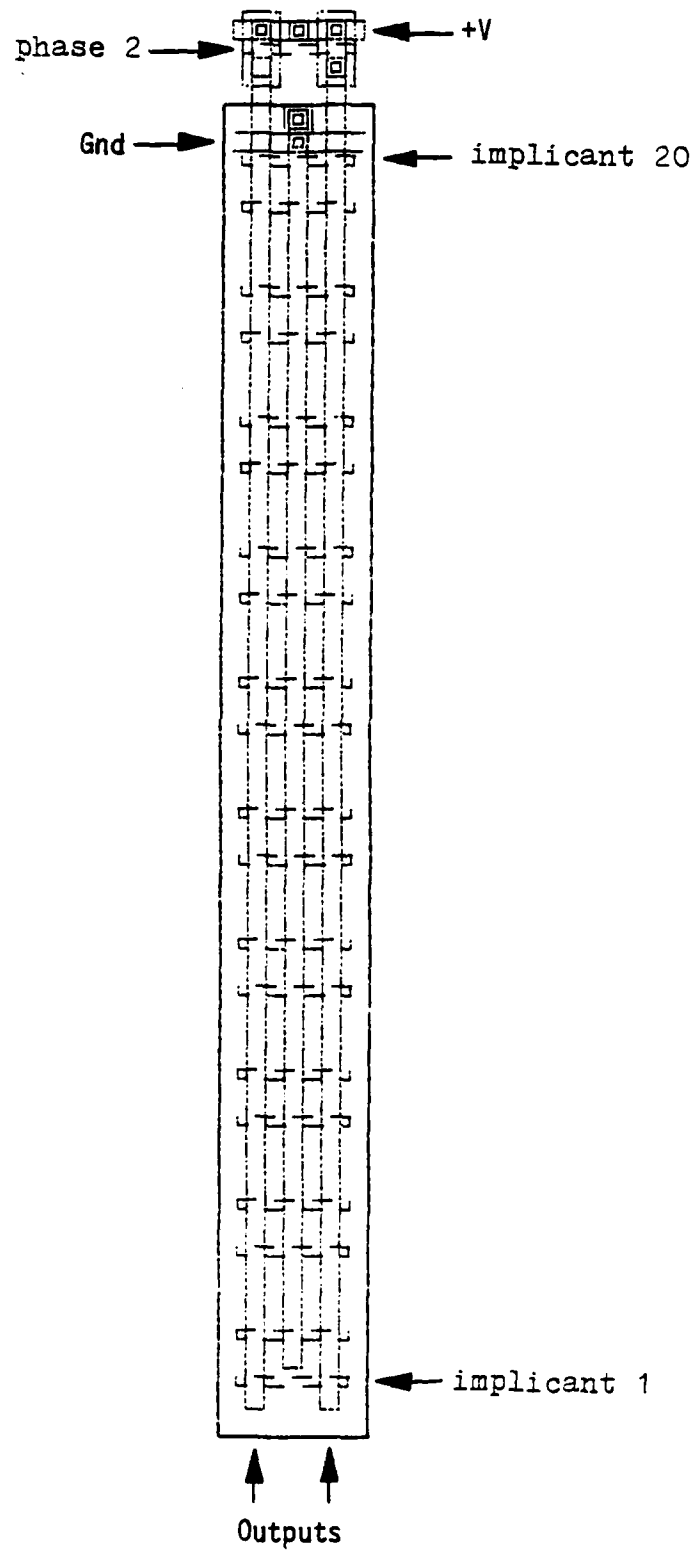
C.8 AND plane circuit for moderate PLA



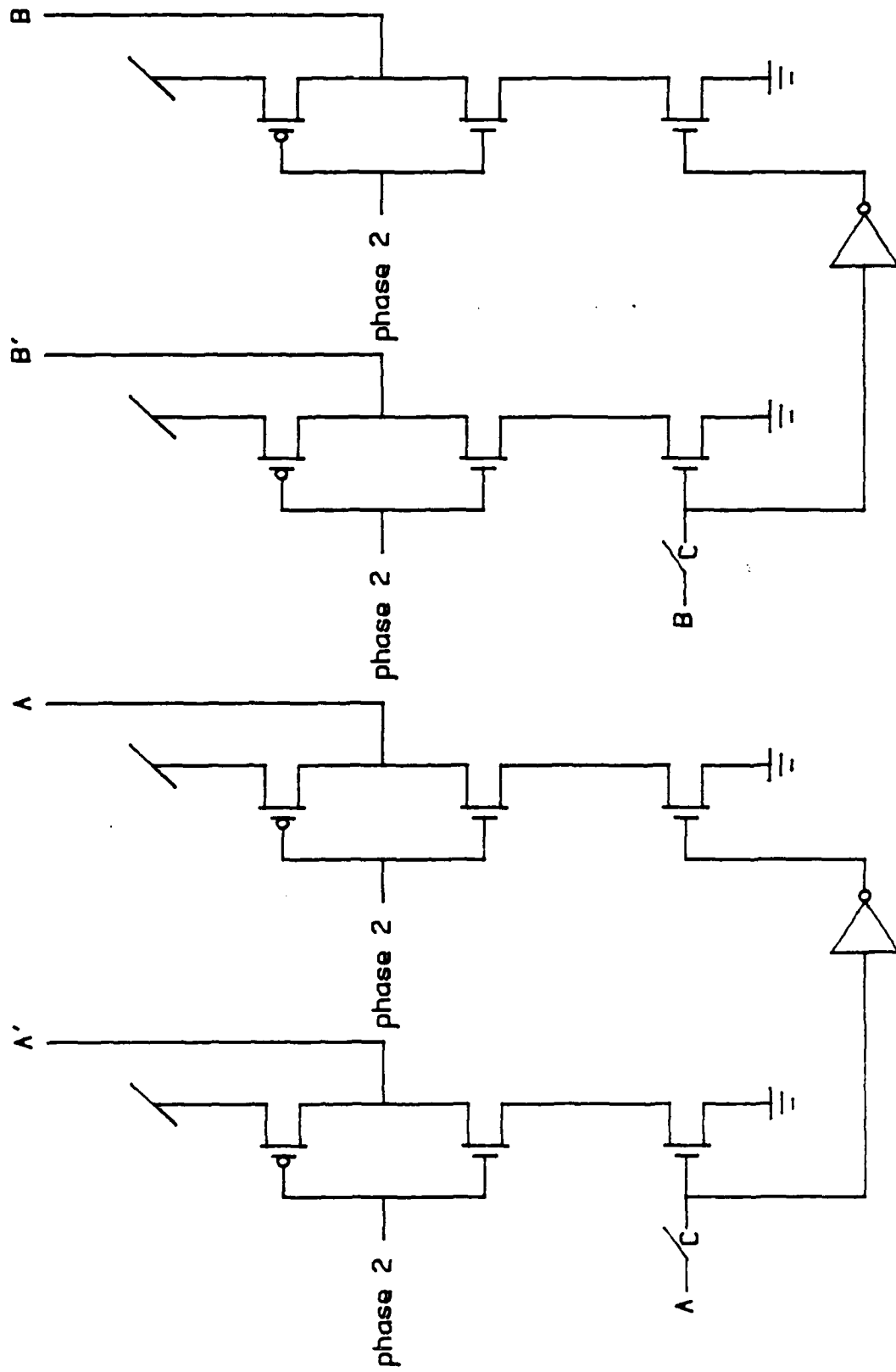
C.8 AND plane circuit for moderate PLA



C.9 OR plane circuit for moderate PLA

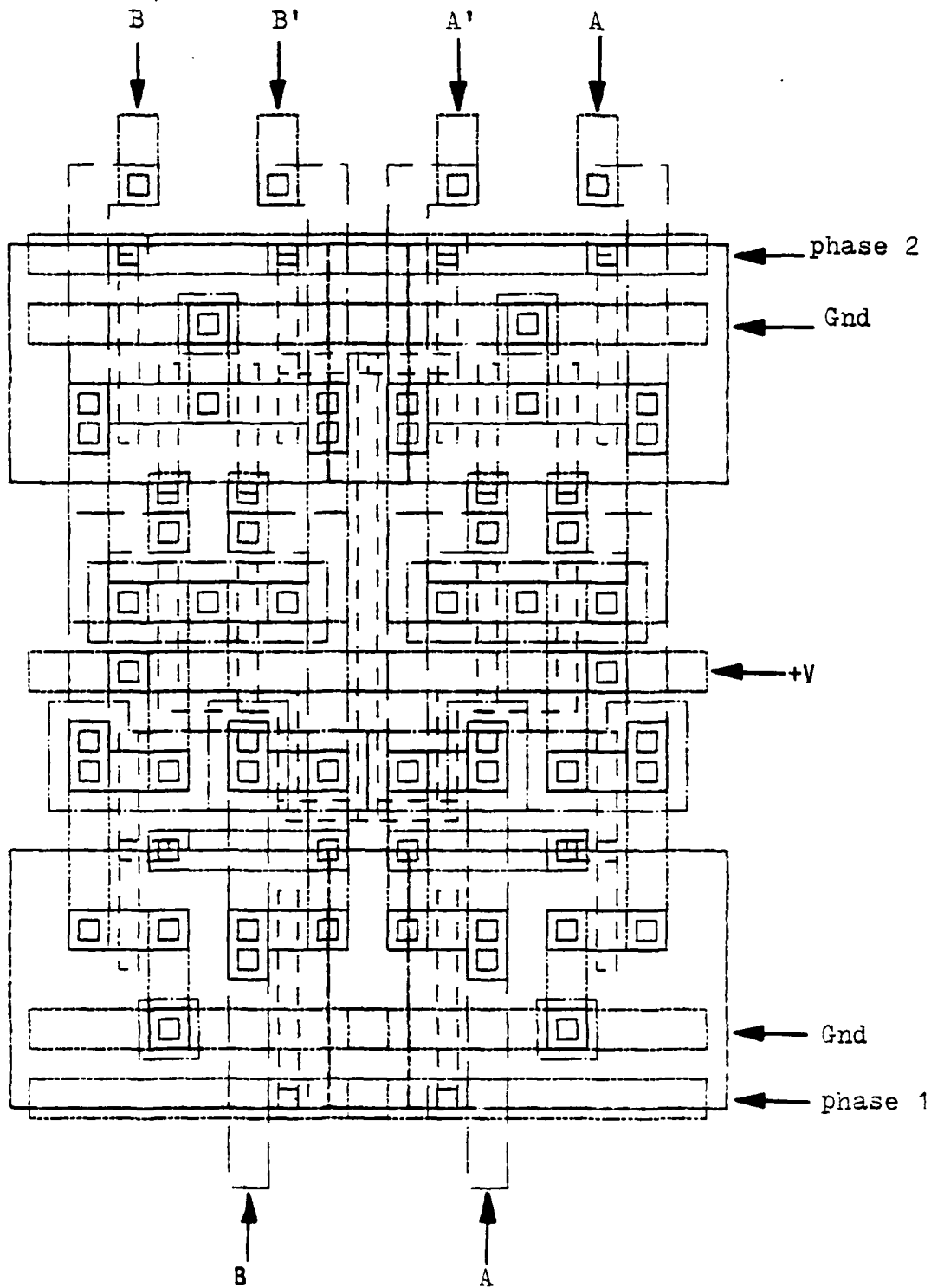


C.9 OR plane circuit for moderate PLA

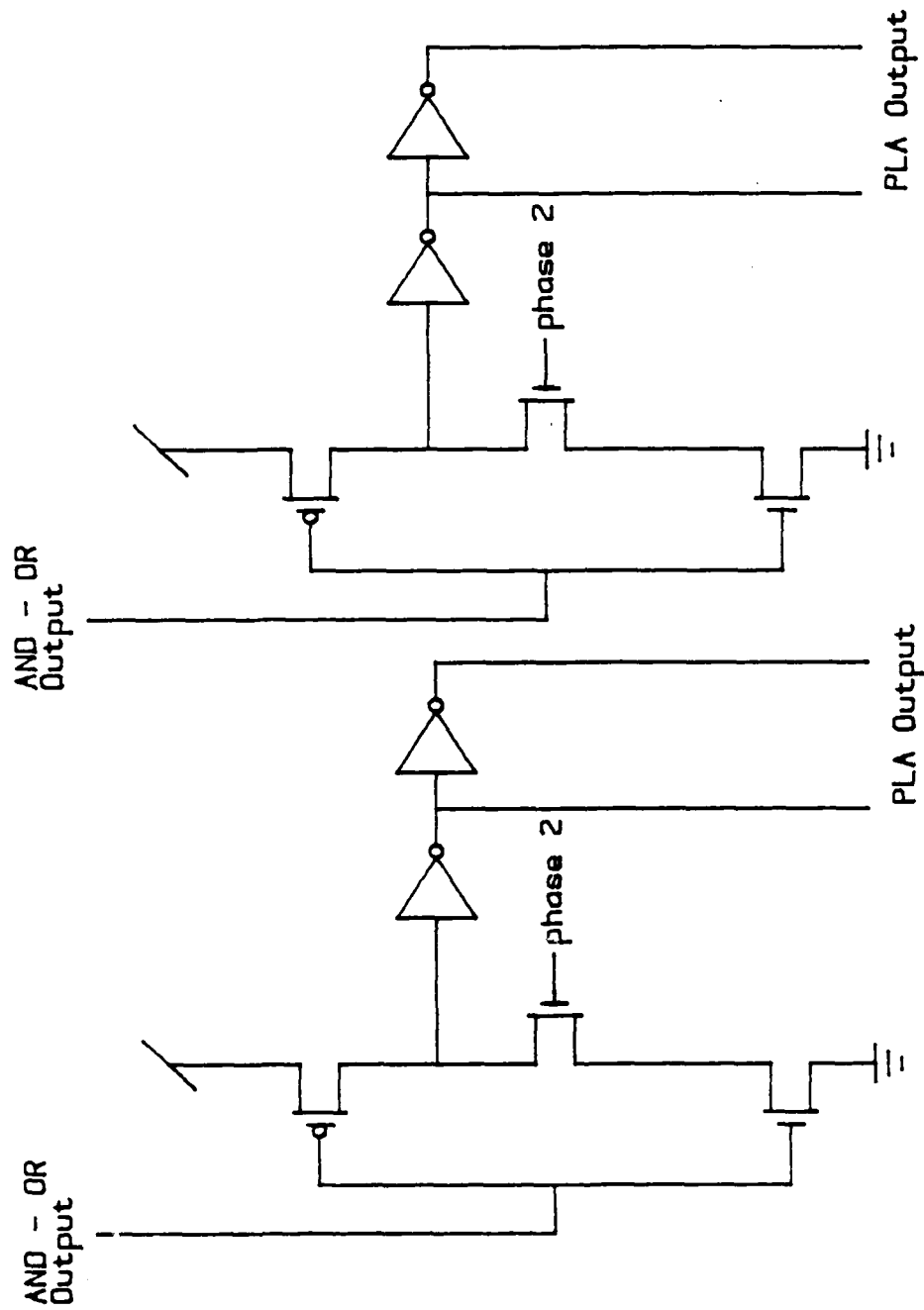


C.10 Input latch for small PLA

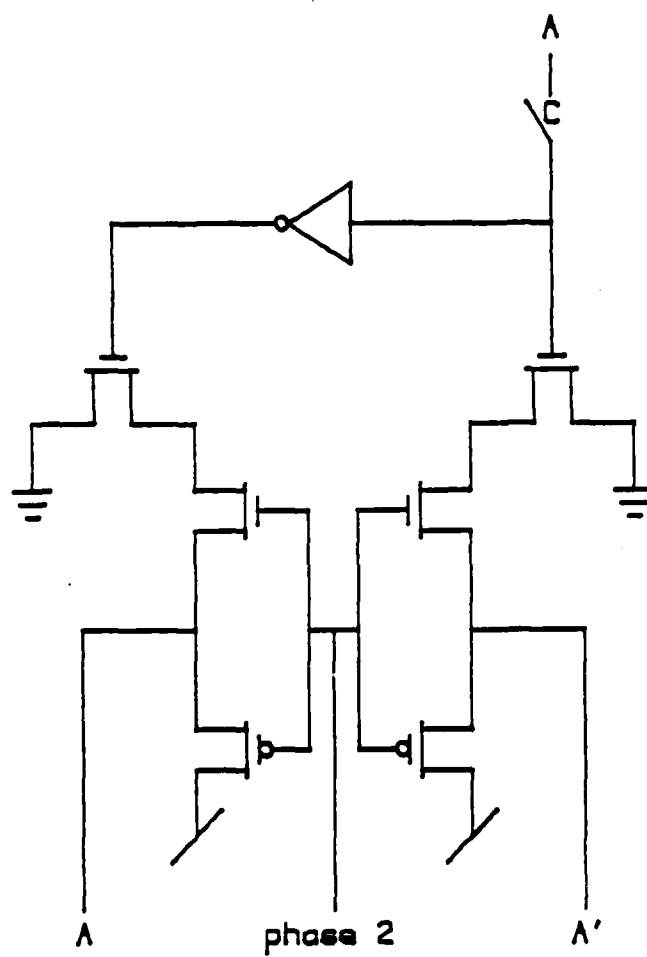




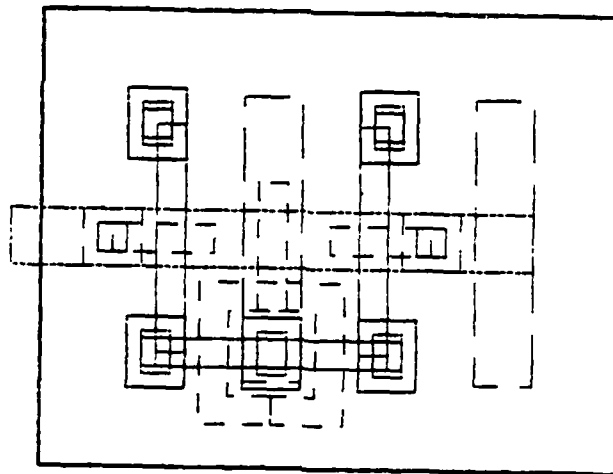
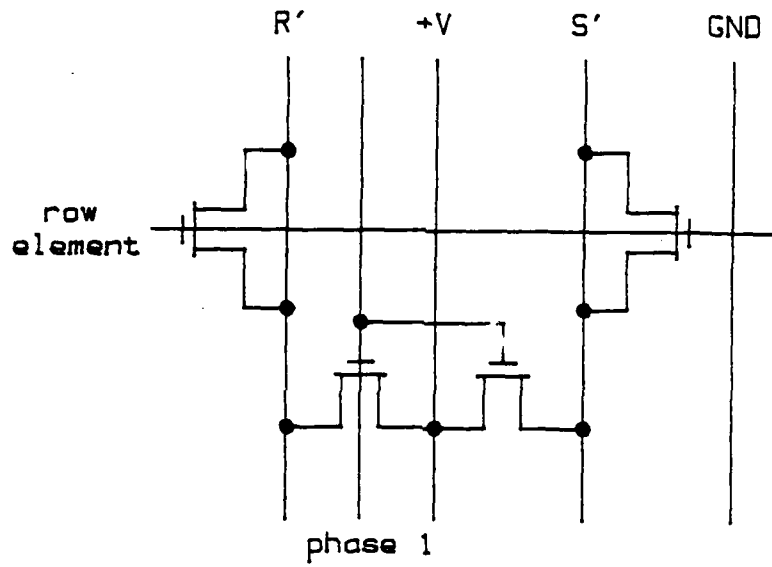
C.10 Input latch for small PLA



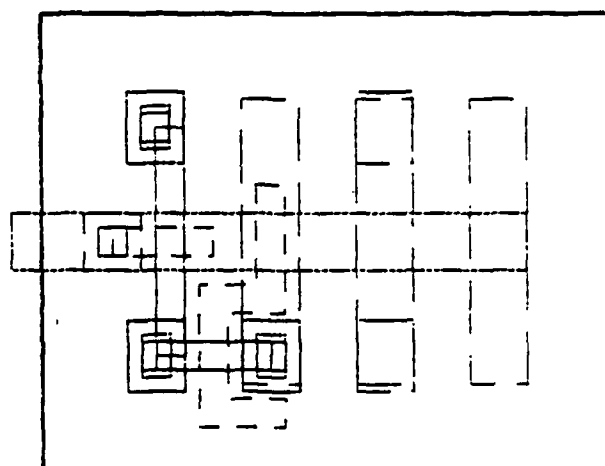
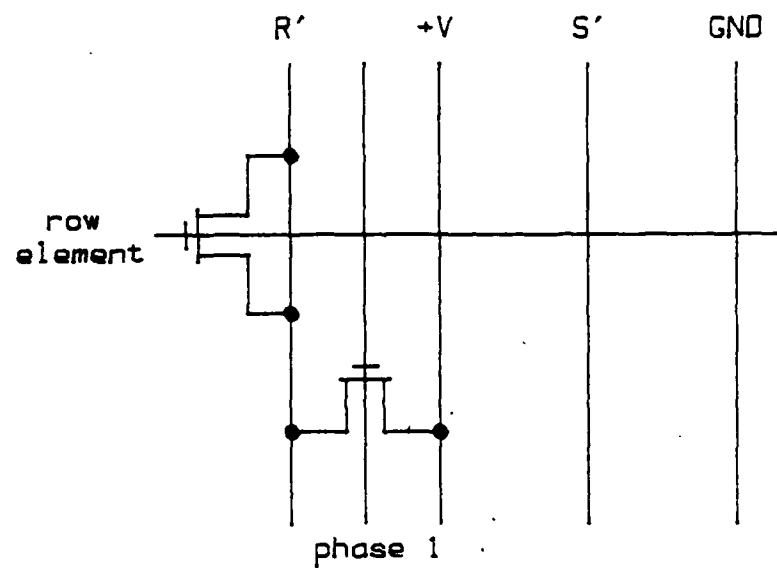
C.11 Output latch for small PLA



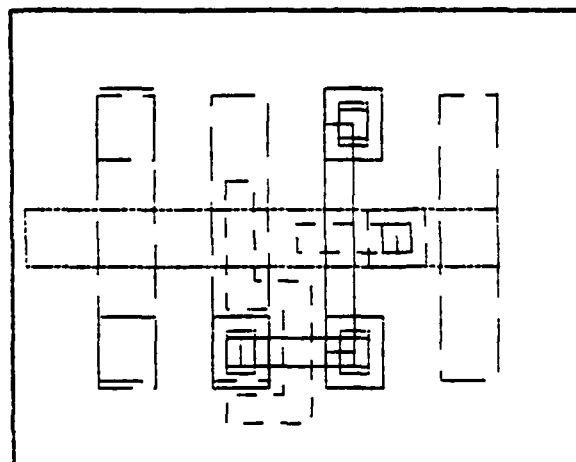
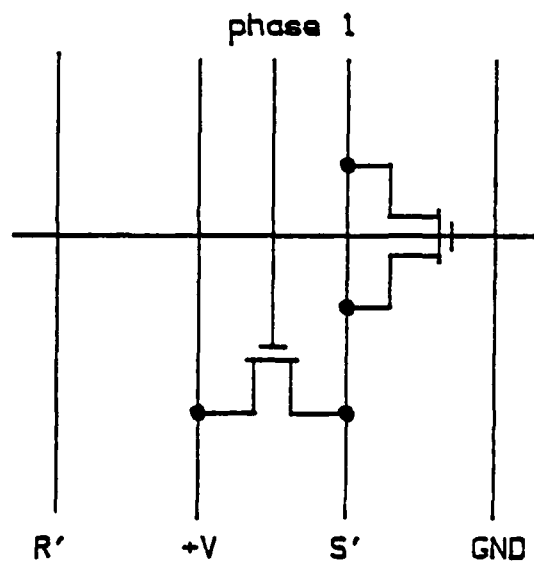
Cell : Input latch  
Symbol : IL  
Size : 13 by 2.5



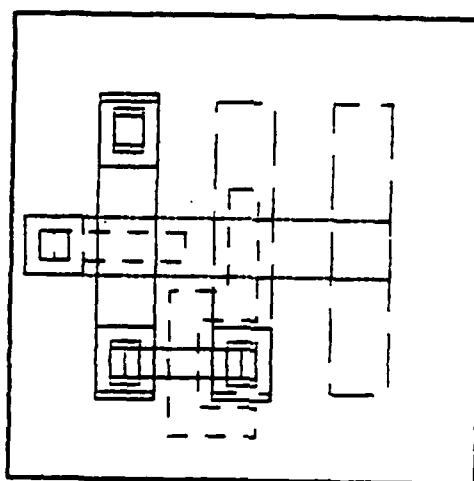
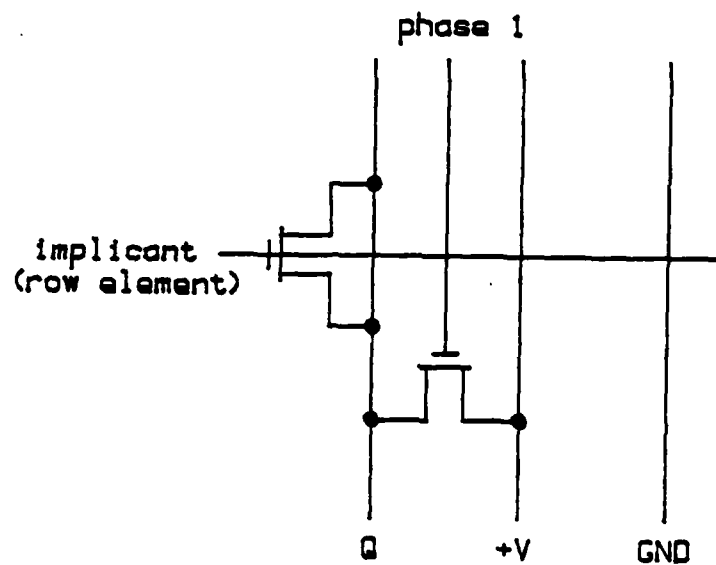
Cell : Set'- Reset'  
 Symbol : S'R'  
 Size : 4 by 2



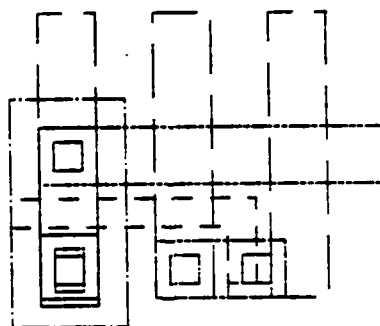
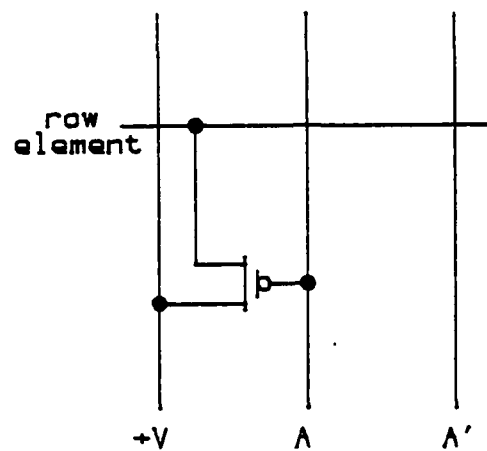
Cell : Reset'  
 Symbol :  $R'$   
 Size : 4 by 2



Cell : Set'  
 Symbol : S'  
 Size : 4 by 2

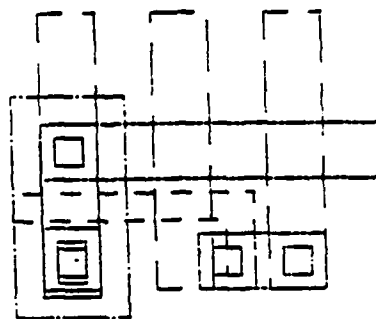
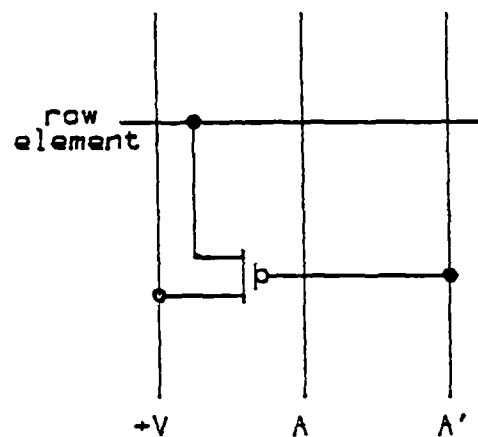


Cell : OR  
 Symbol : +  
 Size : 3 by 2



Cell : One  
 Symbol : 1  
 Size : 2 by 2





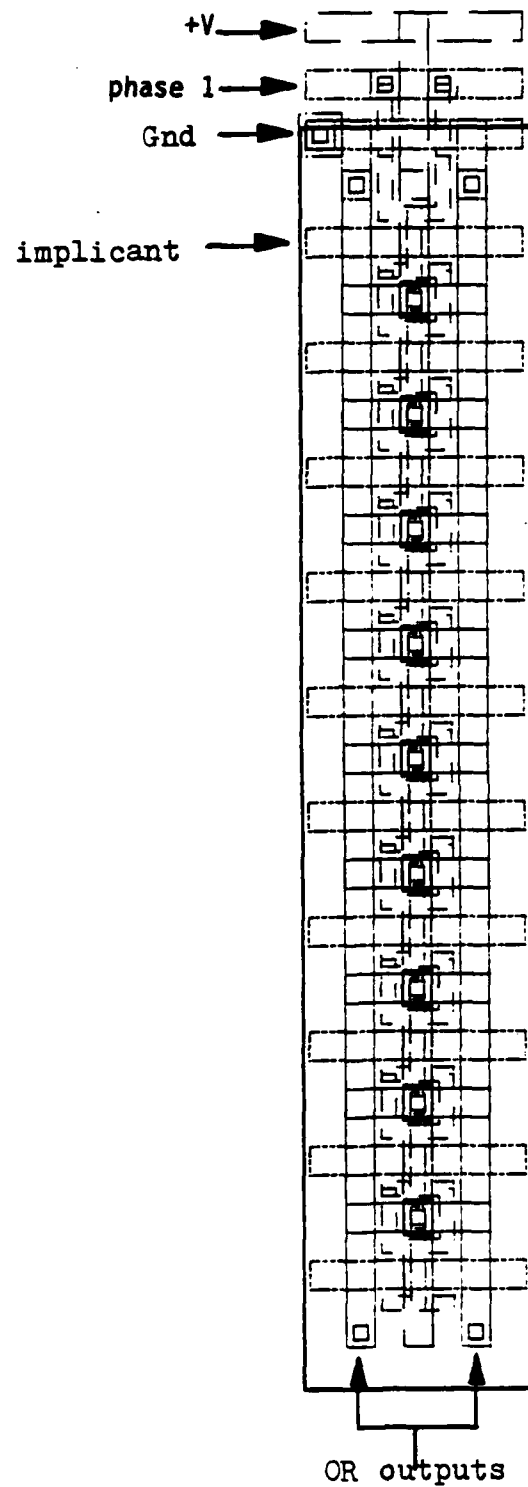
Cell : Zero  
Symbol : 0  
Size : 2 by 2

CONTENTS

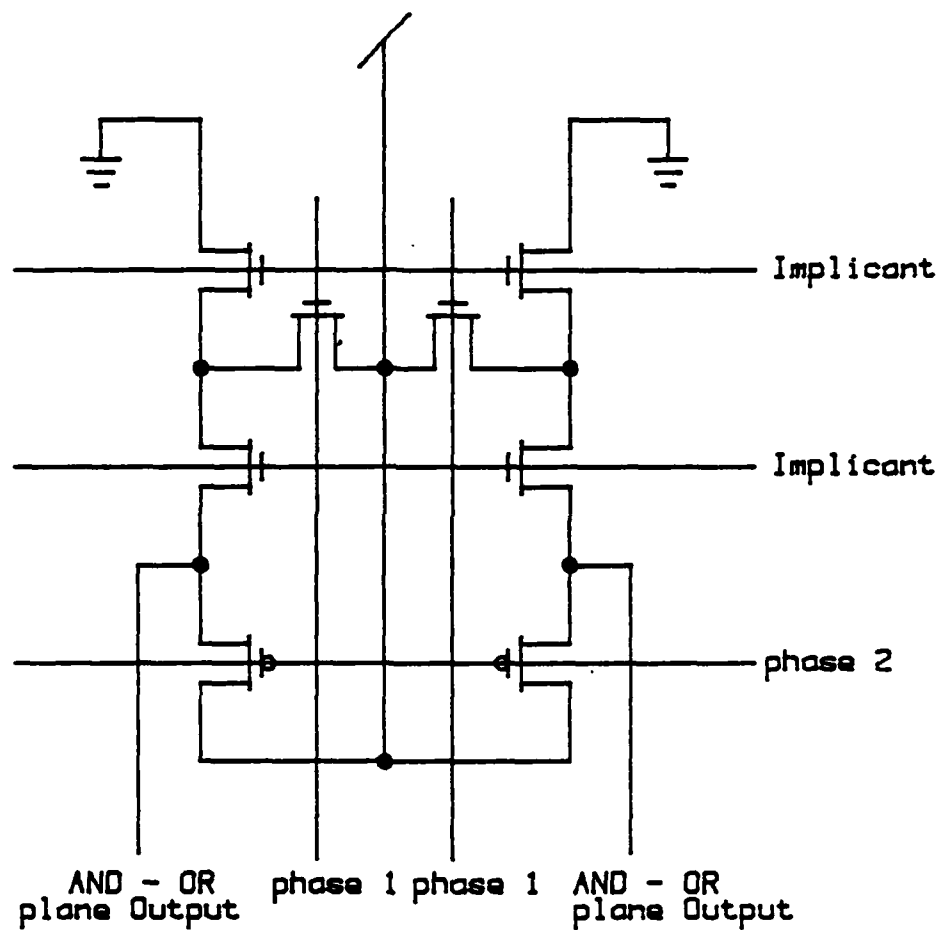
	Page
CELL: Zero . . . . .	161
CELL: One . . . . .	162
CELL: OR. . . . .	163
CELL: Set . . . . .	164
CELL: Reset . . . . .	165
CELL: Set'-Reset' . . . . .	166
CELL: Input latch . . . . .	167-168
CELL: Output latch. . . . .	169-170
CELL: Set-Reset latch . . . . .	171-172
CELL: Precharge row . . . . .	173
CELL: Precharge column. . . . .	174
CELL: Row interconnect. . . . .	175
CELL: Column interconnect . . . . .	176
CELL: Set-Reset column interconnect . . . . .	177

APPENDIX D

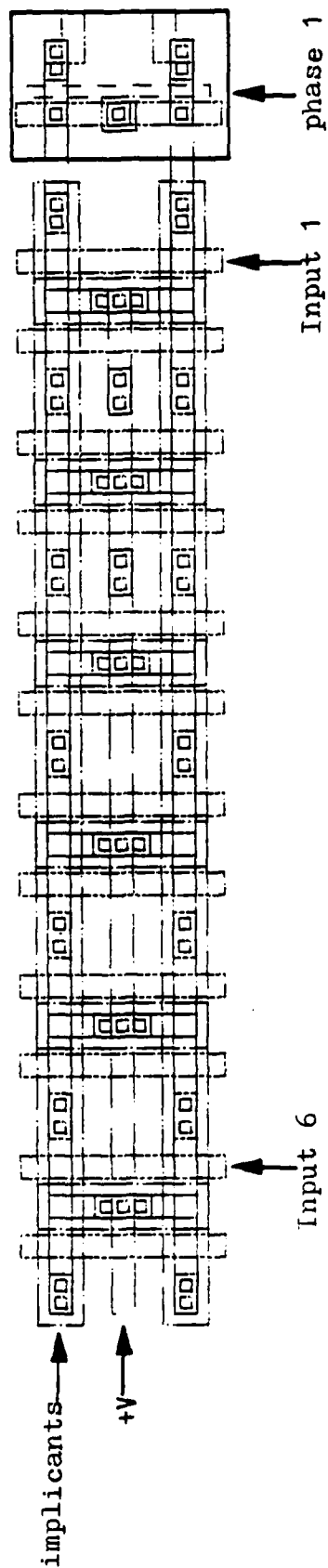
PPL CELL SET



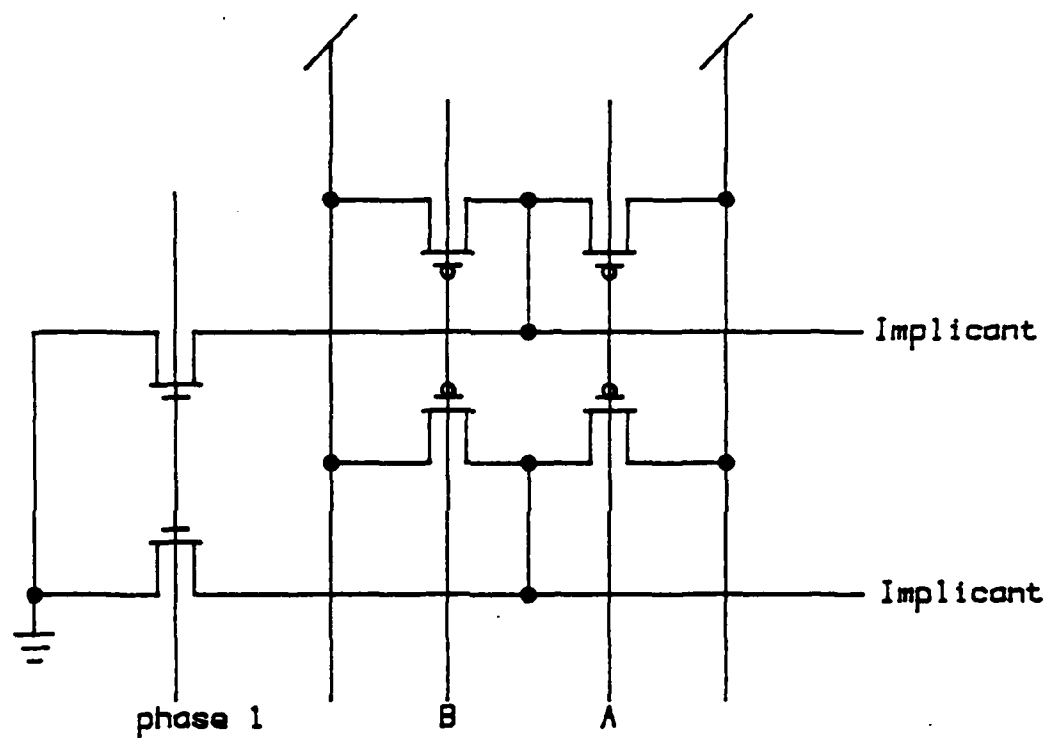
C.13 OR plane circuit for small PLA



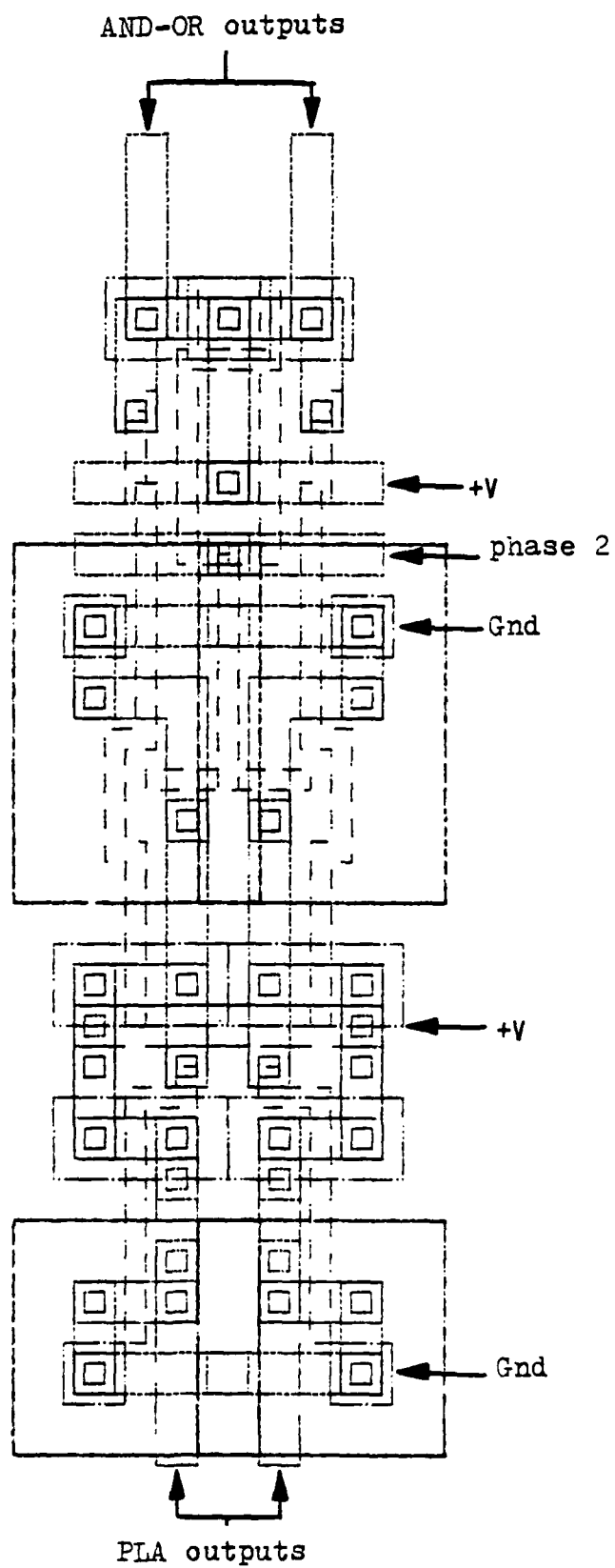
C.13 OR plane circuit for small PLA



C.12 AND plane circuit for small PLA

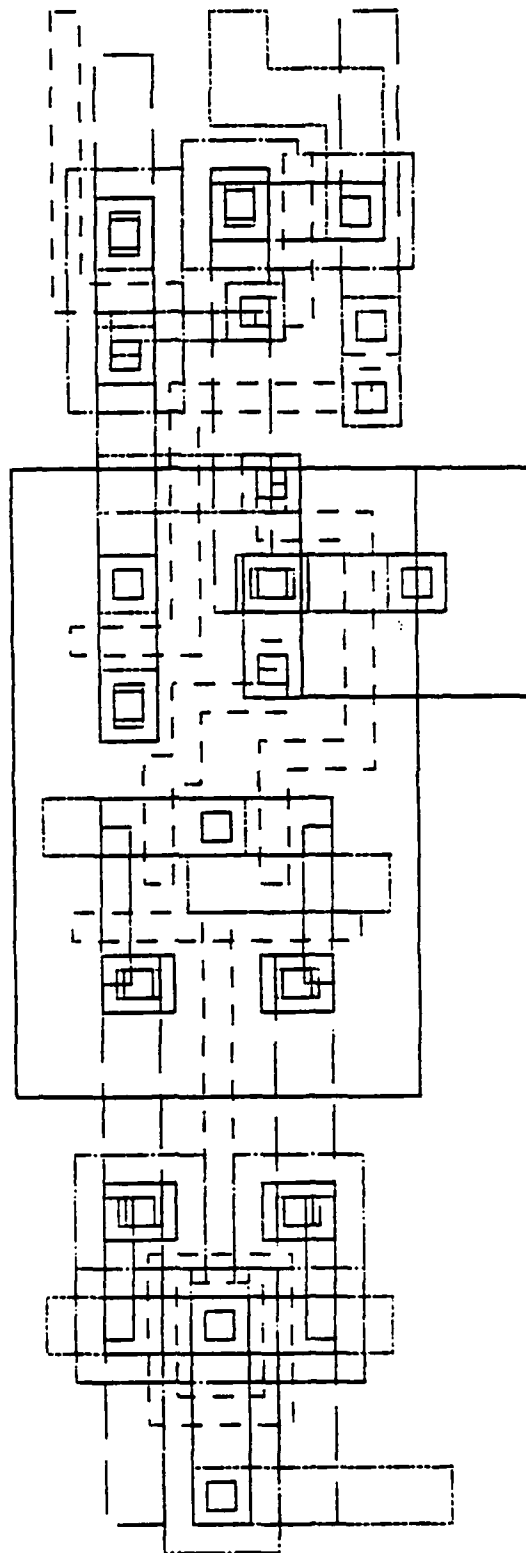


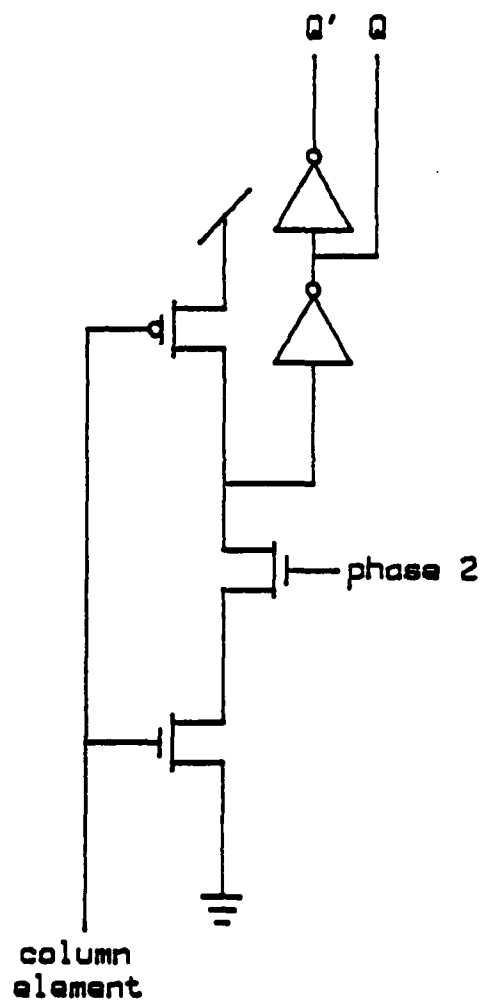
C.12 AND plane circuit for small PLA



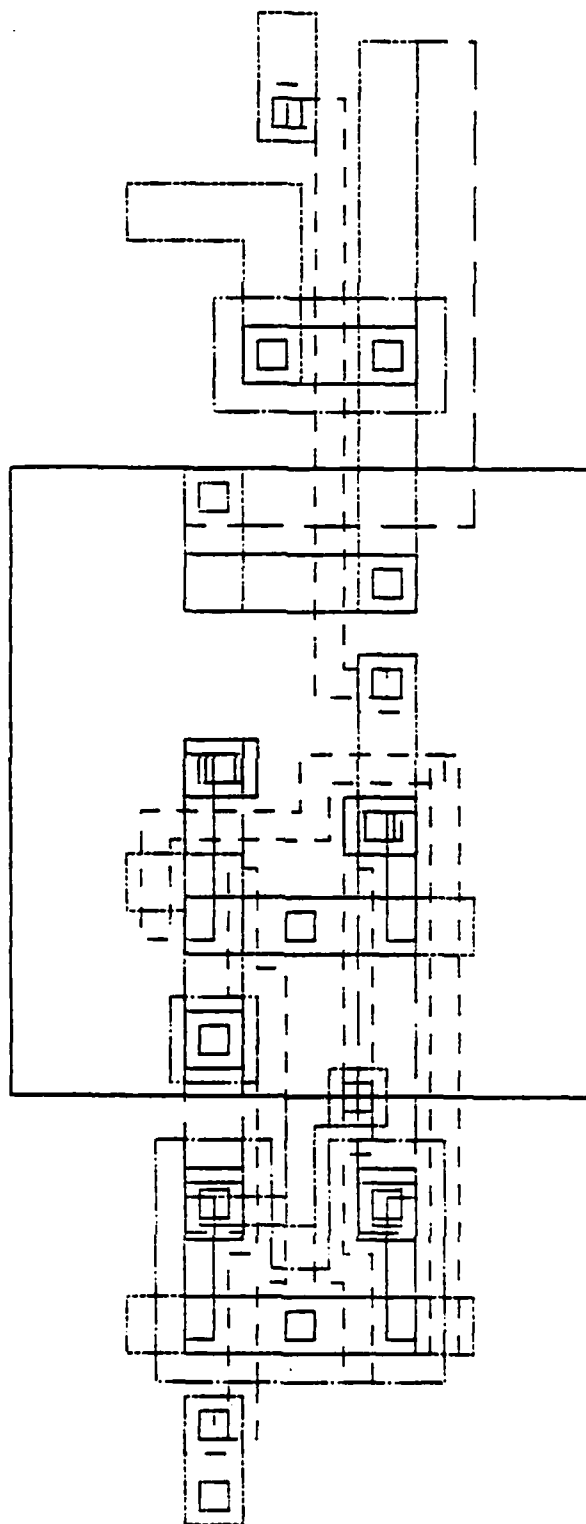
C.11 Output latch for small PLA

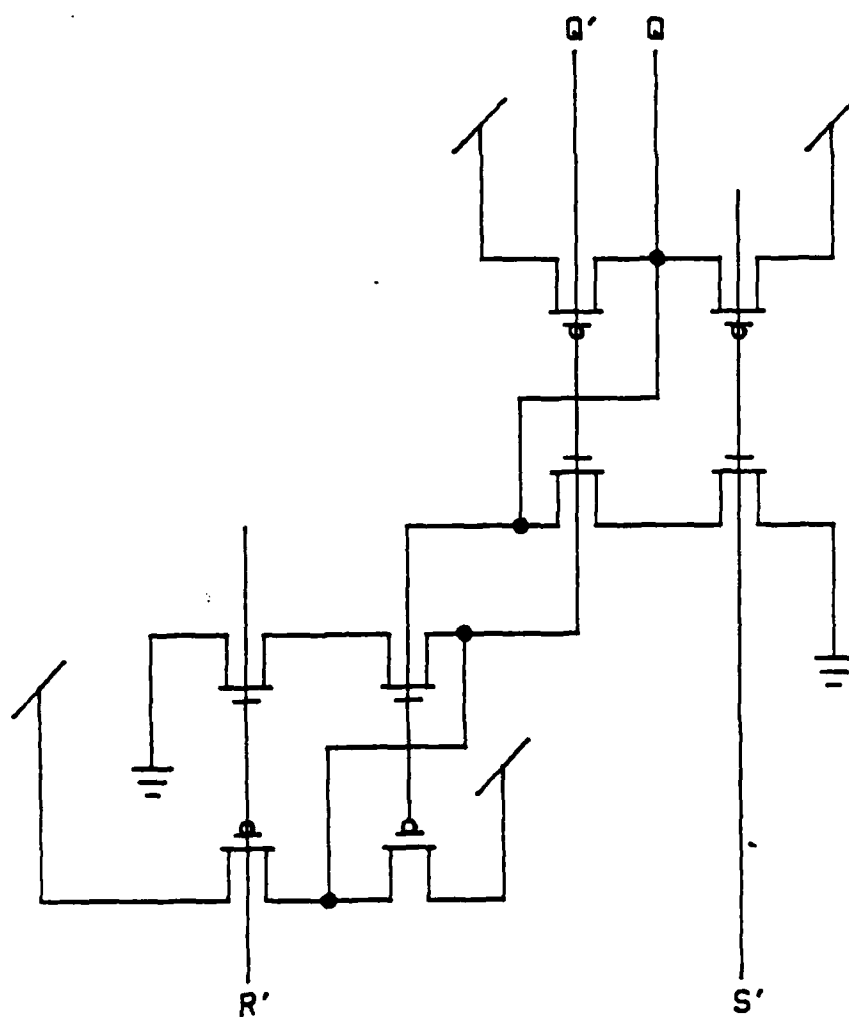




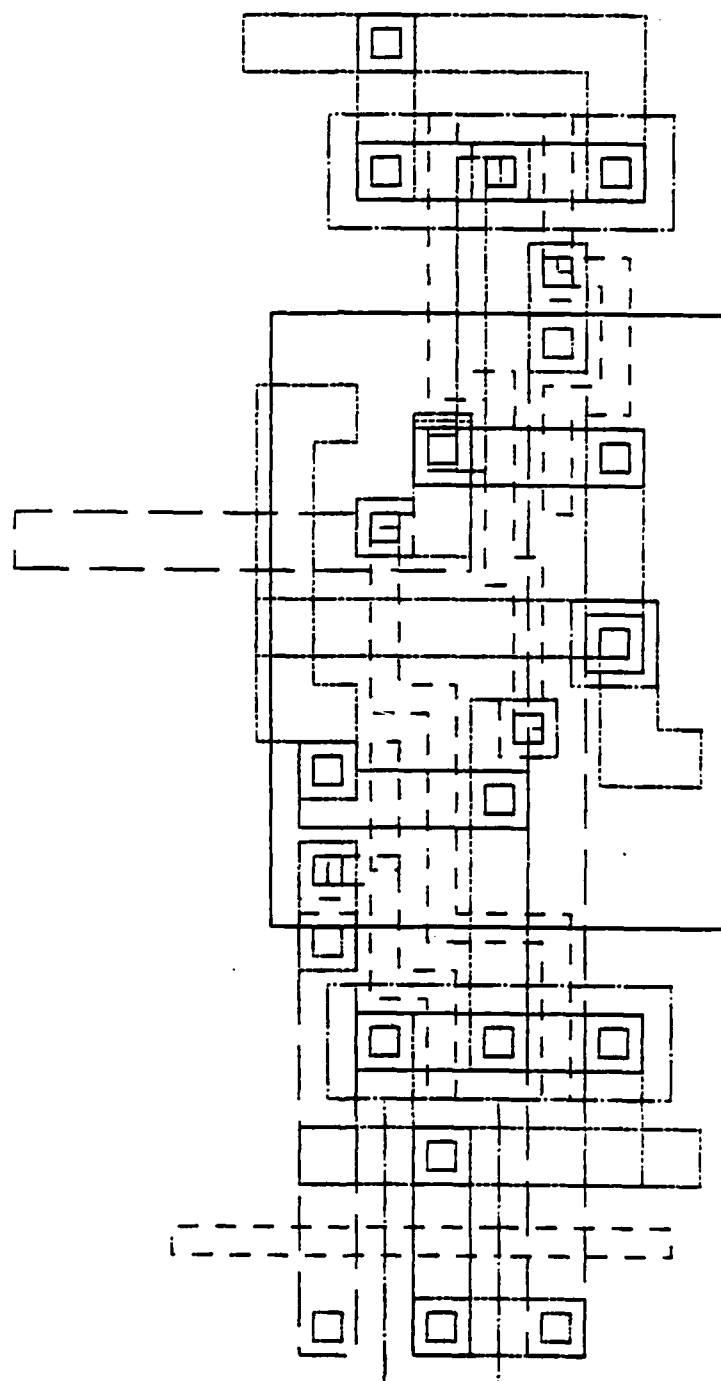


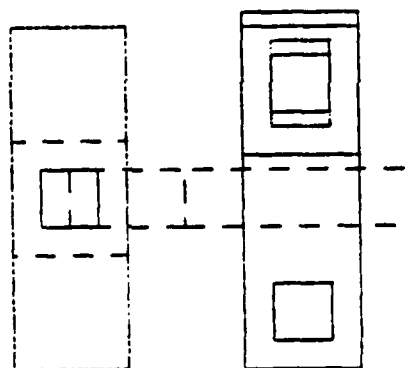
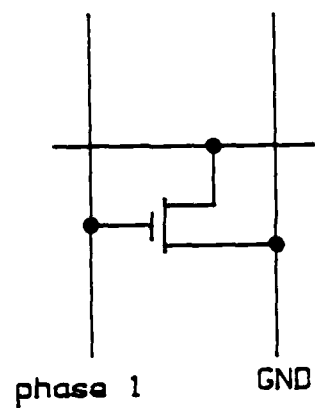
Cell : Output latch  
Symbol : OL  
Size : 13 by 2.5



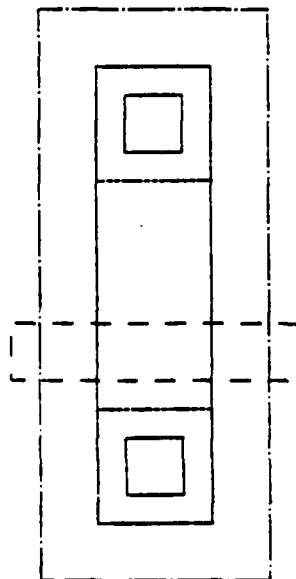
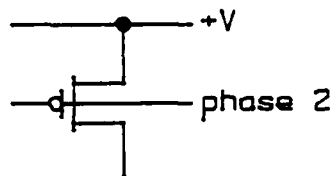


Cell : Set - Reset latch  
Symbol : FF  
Size : 12 by 3.5

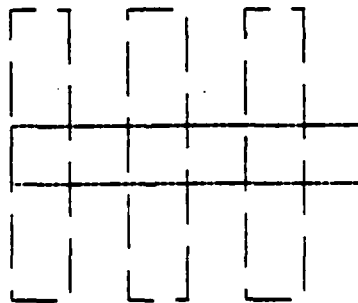
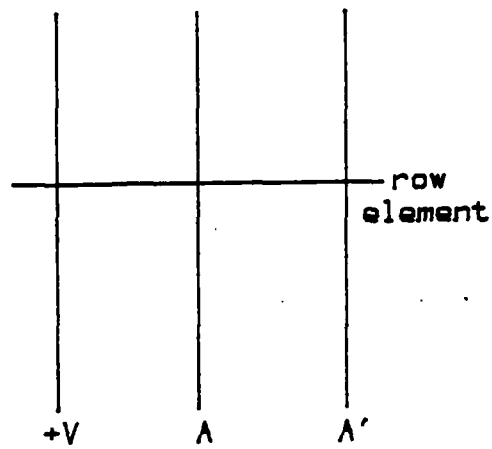




Cell : Precharge row  
Symbol : PR  
Size : 1 by 1

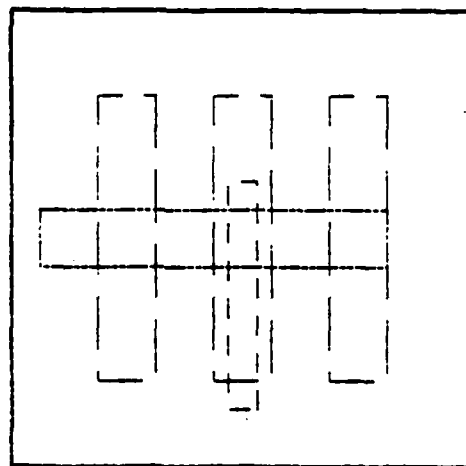
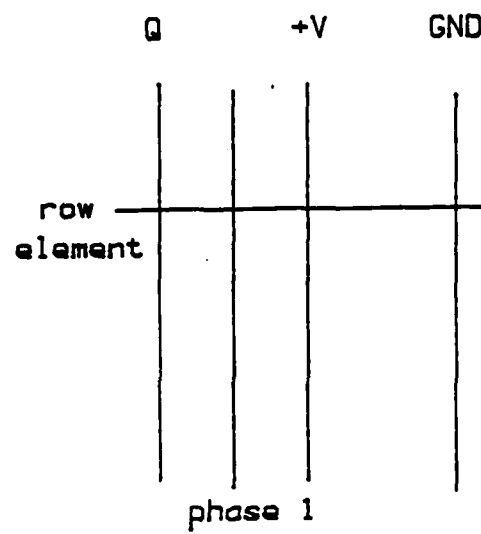


Cell : Precharge column  
Symbol : PC  
Size : 1 by 1

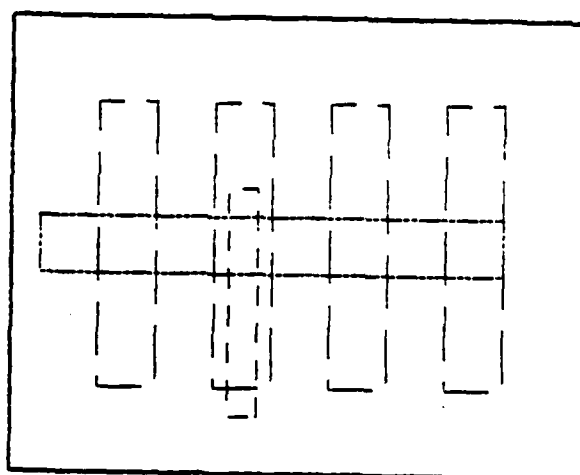
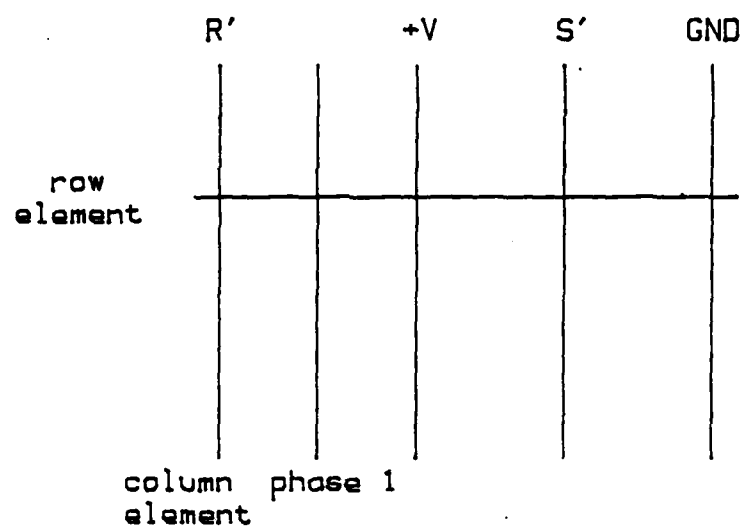


Cell : Row interconnect  
Symbol : RI  
Size : 2 by 2





Cell : Column interconnect  
 Symbol : CI  
 Size : 3 by 2



Cell : Set - Reset column interconnect  
 Symbol : SRI  
 Size : 4 by 2

BIBLIOGRAPHY

1. S. S. Patil and T. A. Welch, "A programmable logic approach for VLSI," IEEE Transactions on Computers, Vol. C-28, pp. 594-601, Sept. 1979.
2. R. A. Wood, "A high density programmable logic array chip," IEEE Transactions on Computers, Vol. C-28, No. 9, pp. 602-608, Sept. 1979.
3. J. D. Trotter, "Stored Logic Arrays - SLAs implemented with clocked CMOS," MSFC, Alabama.
4. D. L. Ostapko and S. J. Hong, "Fault analysis and Test generation for Programmable Logic Arrays (PLA's)," IEEE Transactions on Computers, Vol. C-28, No. 9, pp. 617-627, Sept. 1979.
5. Kent F. Smith, Tony M. Carter and Charles E. Hunt, "Structured Logic Design of Integrated Circuits Using the Storage/Logic Array (SLA)," IEEE Transactions on Electron Devices, Vol. ED-29, No. 4, pp. 765-775, 1982.
6. C. Mead and L. Conway, Introduction to VLSI systems, Massachusetts: Addison-Wesley Publishing Company, 1980.
7. J. Mavor, M. A. Jack and P. B. Denyer, Introduction to MOS LSI Design, Addison-Wesley Publishing Company.
8. David A. Hodges and Horace G. Jackson, Analysis and Design of Digital Integrated Circuits, McGraw-Hill Book Company.
9. Shiva P. Gowni, "Design of a CMOS Microsequencer," M.S. Thesis, Department of Electrical Engineering, Mississippi State University, May 1985.

**END**

**FILMED**

**10-85**

**DTIC**